Titre: Title:	Optimisation et simulation pour la gestion de disponibilite sous comportement d'achat
Auteur: Author:	Thibault Barbier
Date:	2018
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Barbier, T. (2018). Optimisation et simulation pour la gestion de disponibilite sous comportement d'achat [Thèse de doctorat, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/3290/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/3290/
Directeurs de recherche: Advisors:	Gilles Savard, Miguel F. Anjos, & Fabien Cirinei
Programme: Program:	Doctorat en génie industriel

UNIVERSITÉ DE MONTRÉAL

OPTIMISATION ET SIMULATION POUR LA GESTION DE DISPONIBILITE SOUS COMPORTEMENT D'ACHAT

THIBAULT BARBIER
DÉPARTEMENT DE MATHÉMATIQUES ET DE GÉNIE INDUSTRIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR
(GÉNIE INDUSTRIEL)
AOÛT 2018

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

OPTIMISATION ET SIMULATION POUR LA GESTION DE DISPONIBILITE SOUS COMPORTEMENT D'ACHAT

présentée par : BARBIER Thibault
en vue de l'obtention du diplôme de : Philosophiæ Doctor
a été dûment acceptée par le jury d'examen constitué de :

M. BAPTISTE Pierre, Doctorat, président
M. SAVARD Gilles, Ph. D., membre et directeur de recherche
M. ANJOS Miguel F., Ph. D., membre et codirecteur de recherche
M. CIRINEI Fabien, Ph. D., membre et codirecteur de recherche
M. MARCOTTE Patrice, Ph. D., membre
M. ANDERSON Chris K., Ph. D., membre externe

DÉDICACE

À ma famille,

*« It would not be much of a universe if it wasn't home to the people you love. »**

Stephen Hawking.

**L'univers ne serait rien sans les gens que l'on aime.*

REMERCIEMENTS

Ces quatre années de doctorat auront souvent été parsemées de moments de doutes, de stress et puis parfois de réussites. À la fin, l'aventure est belle et ne résulte pas uniquement de mon fait. Je tiens à remercier toutes les personnes qui m'ont accompagné de près ou de loin dans l'aboutissement de mon projet. J'espère n'oublier personne et, si c'est le cas, ce n'est absolument pas volontaire.

Je me dois de commencer par mes directeurs de recherche pour m'avoir accepté, puis encadré et supporté tout au long de mon doctorat. Miguel, pour son soutien, sa présence et ses nombreux conseils qui ont fait de ce doctorat une réussite. Gilles, pour m'avoir fait confiance il y a un peu plus de six ans lors de ma maitrise et pour avoir, à ma plus grande réjouissance, continué l'aventure. Fabien, que j'ai eu la grande chance de pouvoir côtoyé presque tous les jours pendant ces quatre années. J'ai appris tant de choses à ses côtés que j'en découvre encore aujourd'hui les fruits. Plus que cela, il est la parfaite illustration de ce qu'il appellerait une belle personne.

Je remercie Pierre Baptiste et Sébastien le Digabel respectivement président et membre du jury pour avoir accepté d'examiner cette thèse et pour avoir ainsi contribuer à l'améliorer. Je remercie aussi sincèrement Chris Anderson, pour m'avoir fait l'honneur d'être le membre externe de mon jury malgré un long déplacement depuis [lieu]. Enfin, merci au jury de m'avoir délivré mon titre de doctorat.

À propos de l'examen de mes travaux, je ne saurai faire de louanges suffisantes sur le travail de Julie qui, en plus d'éditer parfaitement mon anglais, a rendu mes articles beaucoup plus compréhensibles et cohérents.

Je remercie la compagnie ExPretio Technologies qui m'a fait confiance et a participé au financement de mon doctorat. Au-delà de cet aspect matériel, j'ai eu la chance d'intégrer une formidable équipe dans laquelle je me suis toujours senti intégré et soutenu. Je veux remercier plus particulièrement Kim et Morad de mon équipe de recherche opérationnelle pour tous nos bons moments et nos inspirantes discussions.

Je remercie le Canada et le Québec pour m'avoir accueilli et donné autant d'opportunités. En particulier le conseil de recherches en sciences naturelles et en génie du Canada (CRSNG) et les fonds de recherche du Québec en nature et technologies (FRQNT) qui ont bien voulu subventionner mes travaux.

Au cours de ce doctorat, j'ai pu rencontrer de nombreux autres doctorants notamment à mon

groupe de recherche du GERAD. Nous avons passé de nombreuses heures ensemble à plancher sur des devoirs, à s'auto-motiver, à parler de nos galères du moment et à décontracter au hockey ou au baby-foot. Cette communauté a toujours été bienfaisante et permet de passer outre la solitude dans nos recherches. J'ai donc une pensée très chaleureuse pour Juan, Adham, Hien, Elspeth, Trish, Fillipo, Jaime, Didac, Vilmar et tous ceux que je côtoyais moins fréquemment.

Cette thèse clos 25 années d'études dont le parcours a été rendu possible grâce à de nombreuses belles rencontres avec des professeurs et professeures du lycée Henry Bergson, des Arts et métiers et de l'école Polytechnique de Montréal. Sans eux, je ne serai pas arrivé là où je suis aujourd'hui.

Le doctorat est certes un choix personnel mais il embarque nos proches pendant des années qui ne sont pas toujours évidentes. Ce n'est pas du luxe d'être entouré de personnes capables de vous faire oublier toute pensée ou réflexion sur votre thèse, alors que celle-ci a tendance à revenir rapidement en tête. Ces mêmes personnes ont aussi été capables de me remotiver et même de suivre les avancées avec intérêt. Un grand merci à ma deuxième famille de Montréal : Audrey, Aymeric, Fabien, Kelly, Joris, Laura, Marie-Christine, Marion M, Marion T, Mathias, Mélanie, Thomas et parmi eux mes supers anciens colocs Florian, Nicolas, Sylvie et Sam. Je pense aussi fortement à mes amis d'enfance Florian, Mathieu, Nicolas, Philippe, et Sébastien ainsi que leur compagnes et enfants. A mes compagnons Gadzarts que sont Carole, Benoît, Clément et Guillaume. Et à mes amis rencontrés au Canada que sont Adeline, Corentin, Anaëlle et mon ancien super-coloc Hadrien.

Je ne pourrais continuer ces remerciements sans penser à ma famille dont l'affection, l'amour, le soutien et l'encouragement constants m'ont été d'un grand réconfort et ont contribué à l'aboutissement de ce travail. Un grand merci à mes grands-parents, oncles, tantes et cousins ! Je pense particulièrement à mes grands-parents, Céline et Paul, qui ne cesseront jamais de m'inspirer et de me guider.

Des profonds remerciements vont à mes parents Françoise et Philippe pour m'avoir toujours soutenu, encouragé et aidé. Ils ont su me donner toutes les chances pour réussir. Qu'ils trouvent, dans la réalisation de ce travail, l'aboutissement de leurs efforts ainsi que l'expression de ma plus affectueuse gratitude.

Je remercie mes soeurs Manon et Chloé ainsi que mon frère Titouan pour tout ce qu'ils m'apportent jour après jour et pour notre relation qui m'est inestimable. Ils sont un pilier de ma réussite et à ma joie de vivre. J'y inclus aussi Aniruth, mon merveilleux beau-frère.

Enfin, je suis plus qu'ému au moment de remercier Audrey qui a accepté de devenir ma femme au cours de ce doctorat. Son amour et notre bonheur m'ont été plus qu'essentiels durant ces années. Ce doctorat est aussi le sien tant elle m'a supporté et encouragé.

RÉSUMÉ

Nous nous intéressons dans ce doctorat à l'optimisation de la disponibilité de l'offre au cours d'une période de réservation pendant laquelle des ressources périssables sont vendues. Cette problématique appartient au domaine de la gestion de revenu plus communément désigné par le terme anglais de *revenue management*.

Afin d'illustrer notre propos, considérons le siège d'un train effectuant un trajet entre deux villes à une certaine date et heure. Ce siège est une ressource périssable, car il ne peut pas être proposé une fois le train parti. Cette ressource doit donc être vendue avant son terme durant une période de réservation. Des clients arrivent pendant cette période pour éventuellement réserver cette ressource en achetant un des produits offerts. Les produits sont définis à l'avance par leur tarif et leurs conditions. Ces dernières portent sur l'annulation, l'échange, l'accès au wifi ou une offre de repas par exemple. Chaque client choisira éventuellement d'acheter l'un de ces produits en fonction d'un comportement d'achat qui lui est propre.

Ce doctorat se résume à déterminer quels produits offrir à chaque arrivée de client afin de maximiser le revenu généré par l'ensemble des ventes réalisées au cours de la période de réservation. Pour se faire, les ressources et produits sont fixés ainsi que la demande qui est connue au moyen d'une prévision. L'optimisation des ressources et produits ainsi que la prévision de la demande ne font pas l'objet de ce doctorat et sont d'ailleurs presque toujours traitées séparément dans le domaine de gestion de revenu. Le dilemme suivant est soulevé. D'un côté en acceptant une requête, nous assurons un revenu, mais nous diminuons la capacité disponible pour des requêtes futures éventuellement de meilleur revenu. Il peut donc y avoir de la dilution. D'un autre côté, en refusant une requête, nous réservons de la capacité pour d'éventuelles requêtes futures à meilleur revenu tout en perdant la requête présente à revenu sûr. Il peut donc y avoir du gaspillage.

Deux grandes évolutions ont successivement changé la modélisation de ce problème et ont amélioré la robustesse des solutions tout en le complexifiant. La première a pris en compte les ressources dans leur ensemble plutôt qu'individuellement. La deuxième a intégré le comportement d'achat de clients là où auparavant la demande était considérée de façon indépendante par produit. Aujourd'hui, une formulation exacte intégrant ces deux aspects existe, mais est trop rapidement insoluble. Des approximations ont donc été proposées. L'objectif est de retourner rapidement une politique de disponibilité générant le meilleur revenu espéré une fois simuler.

Dans un premier temps, nous présentons une approximation pour un comportement d'achat

non paramétrique. Ce type de demande est abondamment utilisé en pratique et modélise mieux les substitutions de produits que beaucoup de modèles paramétriques. Il jouit aussi de plus en plus de recherches sur son estimation, mais de peu de travaux en gestion de la disponibilité. Nous introduisons alors un des trois concepts importants de ce doctorat qui est l'utilisation de temps de fermeture par produit comme politique de disponibilité. Cette dernière s'adapte pleinement à la logique d'achat non paramétrique contrairement aux modèles existants. Nous arrivons à un modèle à nombres entiers dont les variables binaires rendent compte d'une hiérarchie de produit qui a un réel sens pratique. Cela nous permet de proposer de bonnes solutions initiales très facilement. La politique par temps de fermeture empêche naturellement la réouverture à la vente de produits au cours de la période de réservation. Il faut savoir que cet aspect est souvent désiré en pratique et nécessite la complexification des approches existantes contrairement à la nôtre. Dans le cas de non-réouverture, nous prouvons que notre approximation est une borne supérieure pour la formulation exacte et qu'elle est asymptotiquement optimale. Nous pouvons utiliser la politique retournée par notre approximation comme solution initiale de n'importe quelle approximation à réouverture. Des résultats numériques sur des instances de petites à grandes tailles montrent que notre approximation, par rapport aux approches existantes, retourne beaucoup plus rapidement une politique de disponibilité générant un revenu espéré légèrement supérieur. Ils mettent aussi en évidence l'accélération des approximations existantes lorsque notre approche est utilisée comme solution de départ.

Dans un second temps, poussés par les résultats précédents, nous cherchons à généraliser l'approche précédente à tout comportement d'achat. Nous représentons d'abord toute demande sous la forme de chemins d'achats formant ainsi un arbre de demande. Cet arbre de demande est le second concept important de ce doctorat. Nous utilisons le fait que chaque chemin d'achat est non paramétrique pour construire une nouvelle approximation acceptant n'importe quel comportement d'achat. Notre nouvelle approximation hérite de la politique de disponibilité par temps de fermeture et donc de la non réouverture. Nous utilisons alors la même linéarisation et nous étendons les résultats théoriques précédents. Les modèles paramétriques retournent un arbre de demande immense qui rend notre approximation insoluble. Pour pallier cela, nous présentons une méthode de résolution itérative basée sur une construction progressive de l'arbre de demande par un ajout successif de chemins d'achats. Nous proposons plusieurs heuristiques pour déterminer quel chemin d'achat ajouter. Chaque ajout raffine la modélisation du comportement d'achat. Nous menons ensuite des expériences numériques sur des instances à comportement d'achat paramétrique de petite à grande taille. Les résultats montrent des résultats similaires à ceux pour le non paramétrique et la méthode itérative de résolution converge vers une bonne solution beaucoup plus rapidement que les

autres approches.

Dans un dernier temps, nous nous penchons sur la simulation pour la gestion de disponibilité. Nous proposons un nouvel estimateur à arrivées fluides pour le calcul du revenu espéré. Cette modélisation par arrivées fluides est le troisième concept important de ce doctorat. Notre modèle agrège alors les différentes arrivées par segment, et ce pour toute la période de réservation. Il ne subit donc pas l'aléatoire d'un ordre d'arrivées. Il nécessite alors une seule évaluation et est invariant alors que l'estimateur traditionnel à arrivées discrètes ne peut réduire la variance et donc augmenter la précision qu'en augmentant le nombre d'évaluations. En contrepartie, nous montrons que notre estimateur présente un biais, contrairement à l'estimateur traditionnel, pouvant être arbitrairement grand même si cela reste minime en pratique. Nous expérimentons alors des méthodes d'optimisation basées sur la simulation afin de résoudre le problème de contrôle de la disponibilité. Nous concluons rapidement que la bonne convergence de ces méthodes dépend largement du point de départ fourni par un modèle en programmation mathématique. Cette conclusion a été un moteur pour le développement des approximations précédentes. D'ailleurs, nous montrons que notre estimateur est équivalent à la plupart des approximations pour le contrôle de la disponibilité. En conséquence, nous évoquons quelques possibilités de notre estimateur pour appuyer l'optimisation. Les résultats numériques sur des instances de grandes tailles témoignent de la nette supériorité de notre estimateur en termes de temps de calcul. Nous constatons aussi peu de biais pour toutes les instances étudiées.

ABSTRACT

This thesis focuses on the availability policy problem when selling perishable resources during a reservation period. This problem belongs to the revenue management topic.

For example, a seat in a train between two cities at 9am on may 3rd 2018 is a perishable resource because it cannot be sold after the train departure. We must sell this resource through a reservation period before it expires and during which customers arrive to eventually buy a product. Many products can exist and are defined in advance by their terms and rates. For example, tickets for this seat can offer cancellation or exchange. A Meal or a wi-fi access can even be proposed at another cost. Each customer will then choose a product or not according to its own choice behaviour.

This thesis aims to determinate which products to offer at each client arrival in order to maximize the income generated by the sales during the reservation period. Products and resources are fixed and demand is known. The products and resources optimization as well as the forecasting are not tackled in this thesis. Besides, they are almost always treated separately in revenue management. The following dilemma is raised. Accepting a request, provides an income, but removes one capacity for a future and potential higher request. This is called spillage. Whereas denying a request, protects a potential higher income but loses an immediate and safe income. This is named spoilage.

Two major developments successively changed the model of this problem and improved the robustness of solutions while increasing the complexity. The first was to consider resources together rather than individually. The second is the integration of the customer choice behaviour rather than an independent demand by product. Today an exact formulation integrates both aspects but is too rapidly intractable. Approximations have therefore been proposed. The challenge is to quickly return an availability policy generating the best expected revenue when simulated.

In a first part, we present an approximation for a non-parametric choice behaviour. This type of demand is widely used in practice and better accounts for products substitutions than many parametric models. It also benefits from recent research on its estimation and is rarely investigated for the availability policy problem. This approximation benefits from one of the three major concepts of this thesis which is an availability policy by closing times per product. This policy fully adapts to the non-parametric buying logic contrarily to existing models. The model is mixed integer with variables that account for a product hierarchy having a practical meaning. This allows us to easily offer good initial solutions in order to

accelerate the resolution of our approximation. The closing time policy naturally prevents the reopening of products during the booking period. This aspect is often desired in practice. To force no reopening, existing approximations must be adapted and are thus more complex to solve. In the case of no reopening, we prove that our approximation is an upper bound for the exact formulation and that it is asymptotically optimal. We can use the policy returned by our approximation as an initial solution of any reopening approximation. Numerical results on small to large instances show that our approximation, compared to existing approaches, returns much faster an availability policy generating slightly higher expected revenue. They also highlight the acceleration of existing approximations when our approach is used as a starting solution.

In a second part, driven by the previous results, we seek to generalize the previous approach to any choice behavior. We first represent any demand in the form of buying paths forming a demand tree. This demand tree is the second important concept of this thesis. Each buying path is non-parametric so that we extend previous results by introducing new approximation accepting any choice behavior. It inherits the closing time availability policy and by consequence the no reopening. We then use the same linearization and extend the previous theoretical results. Parametric choice models return huge and intractable demand tree. To overcome this, we present an iterative resolution method based on a progressive construction of the demand tree by a successive addition of buying paths. We propose several heuristics to determine which buying path to add. We then conduct numerical experiments on small to large instances with parametric choice behaviour. The results are similar to those for non-parametric and the iterative method of resolution converges much faster to a good solution than existing approximations.

In the last part, we focus on simulation for the availability policy problem. We propose a new fluid arrivals estimator to determinate the expected revenue. This fluid arrivals aspect is the third major concept of this thesis. It aggregates the different arrivals by segment for the entire booking period. It thus does not suffer from the randomness of ordered arrivals. Consequently, it requires only one evaluation and is invariant whereas the traditional discrete arrivals estimator can only reduce the variance by increasing the number of evaluations. However, we show that our estimator has a bias, unlike the traditional estimator, which can be arbitrarily large even if remains minimal in practice. We then experiment optimization-based simulation methods to solve the availability policy problem. We quickly conclude that the good convergence of these methods highly depends on the starting point provided by a mathematical programming model. This conclusion has been a motivation for the development of the previous approximations. Moreover, we show that our estimator is equivalent to most of these approximations. As a result, we discuss some applications for our estimator to support

optimization. The numerical results on large-scale instances highlight the superiority of our estimator in terms of computation time. We also found only a little bias for all instances studied.

TABLE DES MATIÈRES

DÉDICACE iii

REMERCIEMENTS iv

RÉSUMÉ vii

ABSTRACT x

TABLE DES MATIÈRES xiii

LISTE DES TABLEAUX xvii

LISTE DES FIGURES xix

LISTE DES SIGLES ET ABRÉVIATIONS xxi

LISTE DES ANNEXES xxii

CHAPITRE 1 INTRODUCTION 1

CHAPITRE 2 REVUE DE LITTÉRATURE 5
2.1 Aspect Réseau 5
2.1.1 Modèles à tronçon unique 5
2.1.2 Formulation exacte du modèle réseau (multiples tronçons) 6
2.1.3 Approximations pour le modèle réseau (multiples tronçons) 7
2.2 Aspect comportemental 9
2.2.1 Modélisations du comportement d'achat 9
2.2.2 Formulation exacte 11
2.2.3 Approximations avec comportement d'achat 11
2.3 Optimisation basée sur la simulation 14
2.3.1 Évaluation par simulation 14
2.3.2 Optimisation basée sur la simulation 15
2.3.3 Calcul du gradient sans utilisation du modèle 16
2.3.4 Calcul du gradient avec utilisation du modèle 17

CHAPITRE 3 ORGANISATION DE LA THÈSE 19

CHAPITRE 4 ARTICLE 1: PRODUCT-CLOSING APPROXIMATION FOR NON-PARAMETRIC CHOICE NETWORK REVENUE MANAGEMENT 21
4.1 Introduction 21
4.2 Related literature 23
4.3 Model 25
4.3.1 Dynamic programming formulation 26
4.3.2 Static approximations 27
4.3.3 Dynamic approximations 28
4.4 Closing approximation 28
4.4.1 Buying logic under closing policy 29
4.4.2 Product Closing Program (PCP) 29
4.4.3 Quality of the PCP 30
4.5 Solving the PCP 32
4.5.1 Linearization 32
4.5.2 Use of hierarchy 33
4.5.3 Reopening (CDPC) 34
4.6 Numerical experiments 34
4.6.1 Parallel flights 35
4.6.2 Bus-line instance 37
4.6.3 Airline instance 39
4.7 Conclusion 43

CHAPITRE 5 ARTICLE 2: BUYING GRAPH FOR CHOICE NETWORK REVENUE MANAGEMENT 45
5.1 Introduction and literature review 45
5.2 Overview 47
5.3 Model 48
5.3.1 Notations 48
5.3.2 Exact formulation 49
5.3.3 Approximations 50
5.4 Buying graph approximation 51
5.4.1 Demand tree 51
5.4.2 Buying graph 53
5.4.3 Quality of the BGP 55
5.5 Linearization and iterative resolution 57
5.5.1 Linearization with hierarchy 57

5.5.2 Iterative resolution . . . 59
5.5.3 Progressive buying graph . . . 60
5.6 Numerical experiments . . . 60
5.6.1 Parallel flights . . . 61
5.6.2 Bus-line . . . 63
5.6.3 Airline . . . 66
5.7 Conclusion . . . 68

CHAPITRE 6 ARTICLE 3: FLUID ARRIVALS SIMULATION FOR CHOICE NETWORK REVENUE MANAGEMENT . . . 70
6.1 Introduction and literature . . . 70
6.2 Simulation for the CNRM . . . 73
6.2.1 Definitions . . . 74
6.2.2 Arrivals process . . . 74
6.2.3 Discrete arrivals simulation (DAS) . . . 75
6.3 Fluid arrivals simulation (FAS) . . . 76
6.3.1 Model formulation . . . 77
6.3.2 Discrete changes . . . 77
6.3.3 Bias and properties . . . 79
6.4 FAS and optimization . . . 80
6.4.1 Simulation-based optimization . . . 80
6.4.2 Equivalence to CNRM optimization . . . 81
6.4.3 Optimization support . . . 82
6.5 Computational results . . . 82
6.5.1 Convergence and bias . . . 83
6.5.2 Estimation time . . . 86
6.5.3 Optimization . . . 89
6.6 Conclusion . . . 91

CHAPITRE 7 DISCUSSION GÉNÉRALE . . . 93
7.1 Synthèse des travaux . . . 93
7.2 Limites des solutions proposées et améliorations futures . . . 95

CHAPITRE 8 CONCLUSION ET RECOMMENDATIONS . . . 97

RÉFÉRENCES . . . 98

ANNEXES . . . 105

A.1 Instances 105
A.2 Parallel flights 106
A.3 Bus-line 107
A.4 Airline 108
B.1 Parallel flights 109
B.2 Bus-line 109
B.3 Airline 112

LISTE DES TABLEAUX

Tableau 2.1 Exemple de modélisation de comportement d'achat de type multinomial logit *(MNL)* pour deux segments (H et L) et quatre produits . . 9
Tableau 2.2 Exemple de modélisation de comportement d'achat par liste de préférence *(PL)* 10
Tableau 2.3 Exemple de modélisation de comportement d'achat par ordre de préférence *(PO)* 11
Table 4.1 Segments for parallel flights. 35
Table 4.2 Running seconds CPU and expected revenue E[R] for Parallel flights by simulation. 36
Table 4.3 Running seconds CPU and expected revenue E[R] for Bus-line by simulation. ΔE[R] is the relative difference with respects to OP policy given by the CDLP-OP. Simulation has 1000 evaluations. 38
Table 4.4 Time CPUp to build the OD policy for Bus-line cumulative changes. 39
Table 4.5 Running seconds CPU and expected revenue E[R] for Airline by simulation. ΔE[R] is the relative difference with respects to OP policy given by the CDLP-OP. Simulation has 500 evaluations. 40
Table 4.6 Airline characteristics by number of cities considered. The five initial cities are ATL, LAX, ORD, DFW, and DEN. 42
Table 5.1 Segments for parallel flights with $w_\emptyset = 0.1$. 62
Table 5.2 Running seconds CPU and expected revenue E[R] for the Parallel flights instance 62
Table 5.3 Running seconds CPU and expected revenue E[R] for the Bus-line instance. 64
Tableau A.1 Instances characteristics. [1] five cities. We use the preference list (PL) choice behavior as presented in the article. 105
Tableau A.2 Parallel flights results 106
Tableau A.3 Bus-line results 107
Tableau A.4 Bus-line results 108
Tableau B.1 Approximations results for the Parallel flights instance. 109
Tableau B.2 Approximation-policy results for the Parallel flights instance. 109
Tableau B.3 Approximations results for the Bus-line instance. 110
Tableau B.4 Approximation-policy results for the Bus-line instance. 111
Tableau C.1 Optimal policies for both instances. 114

Tableau C.2 Bus-line with 150 random policies per PC scenario. 115
Tableau C.3 Airline with 50 random policies per PC scenario. 116

LISTE DES FIGURES

Figure 2.1 Techniques d'optimisation basée sur de la simulation 15
Figure 4.1 Buying logic examples for a segment with preference list $X \xrightarrow{.9} Y \xrightarrow{.8} Z$. 29
Figure 4.2 Resources and products for parallel flights. 35
Figure 4.3 Expected capacity factor relative difference ΔE[CF] with respects to CDLP-OP for Parallel flights. 37
Figure 4.4 Markets for Bus-line instance. 38
Figure 4.5 Markets of Airline. The five largest airports are represented in bold. 40
Figure 4.6 Ideal revenue towards time for Airline. 41
Figure 4.7 Running seconds CPU and expected revenue E[R] for Airline with scaled number of cities considered. 43
Figure 5.1 Running instance (RI). 48
Figure 5.2 Demand tree for the RI. 52
Figure 5.3 Buying logic tree for the RI. 53
Figure 5.4 Buying graph for the RI. 54
Figure 5.5 Resources and products for parallel flights 62
Figure 5.6 Markets of the bus-line instance. 63
Figure 5.7 Convergence of different approximations for the Bus-line instance with $w^{\%}_{\emptyset} = 1$. 65
Figure 5.8 Markets of the Airline instance. 66
Figure 5.9 CDLP columns generation and MBGP iterative resolution by arrival ratio ($\widetilde{\text{MBGP}}_{\text{AR}}$) for the Airline instance with $w^{\%}_{\emptyset} = 5$. The time limit is 3 hours. 67
Figure 6.1 Running instance. 74
Figure 6.2 DAS convergence for the RI. 77
Figure 6.3 Determination of second change for RI. 78
Figure 6.4 Determination of third change for RI. 78
Figure 6.5 Markets for bus-line instance. 83
Figure 6.6 Markets for airline instance. 83
Figure 6.7 Expected revenue estimates μ^F and μ^D with respect to the number of evaluations N for the optimal CDLP availability policy. The FAS relative bias $\Delta\tilde{\theta}^F$ and the DAS 95% confidence interval $IC^D_{95\%}$ are also reported. 84

Figure 6.8 Relative bias $\Delta\widetilde{\theta}^F$ with respect to the percentage γ_{close} of closed products. 85
Figure 6.9 Relative bias $\Delta\widetilde{\theta}^F$ for the optimal and random policies when capacity is scaled and demand adjusted proportionally. 86
Figure 6.10 Estimation time for FAS and DAS ($[\text{IC}^D_{95\%}]$ is 5%, 1%, and 0.5%) with respect to γ_{close}. 87
Figure 6.11 Evaluation time CPU/N for the two estimators, averaged over the load factors 0.8, 1.0, and 1.2. 88
Figure 6.12 SP technique applied to the FAS estimator (μ^F). The solutions returned by FAS are simulated by DAS (μ^D). 89
Figure 6.13 Two examples of FAS optimization support in the solution of the CDLP for the airline instance (LF=1). 90
Figure B.1 Time needed to build the OD policy for the Bus-line ($LF = 1$, $w_\emptyset = 1\%$). 110

LISTE DES SIGLES ET ABRÉVIATIONS

GR	*Gestion de Revenu*
GD	*Gestion de la Disponibilité*
BGP	*Buying Graph Program*
CDBG	*Choice Deterministic Buying Graph*
CDLP	*Choice Deterministic Linear Program*
CDPC	*Choice Deterministic Products Closing*
CNRM	*Choice Network Revenue Management*
CPU	*Central Processing Unit*
CRS	*Central Reservation System*
DAS	*Discrete Arrivals Simulation*
FAS	*Fluid Arrivals Simulation*
LBGP	*Linear Buying Graph Program*
MBGP	*Mixed integer Buying Graph Program*
OD	*Offers Duration*
OP	*Offers Period*
PB	*Products Booking*
PC	*Products Closing*
PCLP	*Products Closing Linear Program*
PCMP	*Products Closing Mixed integer Program*
PCP	*Products Closing Program*
RM	*Revenue Management*
SDCP	*Segment-based Deterministic Concave Program*

LISTE DES ANNEXES

Annexe A ARTICLE 1 E-COMPANION 105
Annexe B ARTICLE 2 E-COMPANION 109
Annexe C ARTICLE 3 APPENDIX 113

CHAPITRE 1 INTRODUCTION

Gestion du revenu (GR). La gestion de revenu (GR), ou *revenue managment* en anglais, est l'ensemble des techniques pour la maximisation du revenu de la vente de ressources éventuellement périssables. Le leitmotiv est alors de *vendre le bon produit au bon client et au bon moment*. La GR se décompose en plusieurs problèmes :

La *prévision de la demande* doit prédir la quantité de clients potentiels ainsi que leur comportement d'achat. Plusieurs modèles existent pour modéliser mathématiquement cette logique d'achat. Les plus connus et utilisés sont les modèles indépendants, paramétriques (multinomial logit) et non paramétriques (preference list). La plupart du temps les clients sont divisés par comportement en segments. Pour chaque segment, il s'agit alors d'estimer la quantité ainsi que les paramètres de la modélisation du comportement d'achat. La plupart du temps des méthodes d'apprentissage sont utilisées sur des données de réservations historiques.

L'*établissement des ressources* doit établir l'ensemble des ressources en vue de maximiser le revenu. Pour le domaine ferroviaire ou aérien, cela prend la forme d'un plan de transport. Des trains ou des avions sont alors alloués sur chacun des différents trajets souhaités. Pour le domaine hôtelier, il s'agit par exemple de l'agencement des différents types de lits dans chaque chambre.

L'*établissement des produits* doit fixer les différents produits dans le but du meilleur profit. Les différentes conditions de ventes sont alors décidées ainsi que les tarifs. La combinaison de condition(s) et d'un tarif forme un produit. Cette partie est souvent menée par le département de marketing.

La *gestion de la disponibilité* doit contrôler la disponibilité des produits au cours de la période de réservation dans le but de maximiser le revenu. C'est la partie sur laquelle porte ce doctorat.

Modélisation du problème de GR. Les quatre problèmes précédents de la GR sont liés mais ne sont actuellement pas résolus dans un même modèle pour des raisons organisationnelles, historiques, mais aussi de complexité. Ils sont chacun traités séparément et différemment. Ainsi, la gestion des ressources appartient aux décisions stratégiques. L'établissement des produits et la prévision de la demande correspond aux décisions tactiques. Finalement la gestion de la disponibilité fait partie des décisions opérationnelles. Dans l'état actuel, ces problèmes interfèrent de façon statique alors qu'en réalité ils s'influencent. Par

exemple, La gestion de la disponibilité considère les produits comme fixés et fournis par l'établissement des produits. Aujourd'hui les recherches sur chacun des problèmes de la GR tendent à incorporer plus de réalisme dans les modélisations. Les solutions retournées sont alors plus robustes. La contrepartie est que les modèles sont plus compliqués à résoudre.

Sous-problème de gestion de la disponibilité (GD). Dans ce doctorat, nous traitons uniquement la gestion de la disponibilité et ne développons pas les autres sous-problèmes de la GR. Ce problème a plusieurs désignations dans la littérature mais les travaux les plus récents tendent à s'accorder sur le terme *availibility control problem*. Nous rappelons qu'il s'agit de contrôler la disponibilité des produits ou des ressources au cours de la période de réservation. En contrôlant les ventes tout au long de l'horizon de reservation, nous pouvons maximiser le revenu. Il est central car il prend en entrée les solutions des trois autres problèmes pour pouvoir être résolu. En effet les ressources, produits et demande sont considérés comme fixés et connus pour ce problème. Les recherches se sont d'abord portées sur l'optimisation avec une seule ressource. Puis des modèles plus complexes prenant en compte l'ensemble des ressources ont été développés. Les résultats numériques ont alors montré l'importance de considérer toutes les ressources interdépendantes dans un même modèle de façon à obtenir une solution plus robuste se traduisant en un revenu espéré plus élevé. Cette aspect « réseau » a progressivement été implanté dans tous les travaux. Par la suite, les recherches ont suivi le développement de la modélisation du comportement d'achat pour le problème de prévision de la demande et ont commencé à tenter d'incorporer celui-ci. Les diverses expériences ont vite mis en évidence la supériorité d'une modélisation à comportement d'achat sur le modèle indépendant de demande.

Résolution du problème de GD. Une formulation exacte en programmation dynamique a été établie pour traiter du problème de GD avec comportement d'achat et à multiples ressources. Celle-ci est très rapidement impossible à résoudre car le nombre d'états augmente de facon exponentielle avec le nombre d'arrivées potentielles et la capacité de chaque ressource. De ce fait, les recherches portent sur des approximations permettant de résoudre le problème de GD plus rapidement. Ces approximations sont détaillées dans la revue de littérature suivante. Le but est alors de développer un modèle faisant le moins de compromis possible sur la modélisation du problème tout ayant un temps de résolution acceptable dans la pratique. Un contrôle de disponibilité doit ainsi être retourné dans le temps alloué et doit fournir le meilleur revenu espéré une fois évalué par une simulation. Cette simulation n'optimise pas mais évalue le revenu à espérer. Elle se base donc sur une modélisation très réaliste du problème de GD, contrairement aux approximations utilisées pour l'optimisation. Ces

approximations sont pour la plupart capables d'accepter en théorie n'importe quelle modélisation du comportement d'achat. Cependant pour être résolue efficacement, leurs méthodes de résolution sont souvent spécifiques à une seule modélisation du comportement d'achat. Celui le plus développé aujourd'hui est basé sur un modèle paramétrique. Notamment car il fut le premier à être introduit dans la littérature de l'économie et du marketing. Il attribue un poids représentatif de l'intérêt du client vis à vis de chaque produit. Ces poids permettent de calculer la probabilité d'acheter un produit en fonction de ceux qui sont offerts.

Nouvelle approximation pour le non paramétrique. Aujourd'hui les recherches les plus récentes sur l'estimation de comportement d'achat se tourne vers un autre modèle basé sur des ordonnancements de produits. Cette modélisation non paramétrique ne présente pas certains défauts du paramétrique jusque-là plus largement étudié. Il ne souffre pas, par exemple, de capturer incorrectement les différentes substitutions de produits similaires qui est un défaut connu du paramétrique. Il est de plus déjà utilisé en pratique dans certaines compagnies spécialisées dans la gestion de revenu. Pour le moment, la plupart des approches en gestion de la disponibilité pour le non paramétrique sont des adaptations d'approximations existantes développées pour le paramétrique. La modélisation non-paramétrique comporte des mécanismes logiques qui lui sont propres. Une approximation rendant compte de ces spécificités pourrait alors être beaucoup plus robuste et rapide que les approches actuelles qui ne sont pas forcément adaptées à ce modèle de comportement.

Nouvelle approche itérative pour l'optimisation des plus grosses instances. Les méthodes actuelles les plus adaptées à la résolution d'instances de taille importante pour le problème de GD sont celles qui peuvent se résoudre de manière itérative. Elles retournent ainsi une solution qui s'améliore au fur et à mesure du temps alloué contrairement aux méthodes directes dont la résolution peut durer plus longtemps que la fenêtre de temps à respecter. La génération de colonnes est l'approche la plus adaptée pour le moment pour résoudre efficacement les grosses instances. Cependant elle n'est pas forcément appropriée à tous les modèles de comportement d'achat. D'abord car le sous-problème est NP-hard à résoudre et que sa complexité dépend du nombre de produits. Ensuite car les colonnes générées sont des ensembles de produits qui ne sont pas adaptées au modèle non paramétrique où les chemins d'achat ne nécessitent pas de travailler avec la combinatoire des ensembles de produits. L'utilisation des listes de préférences du non paramétrique permet ainsi de penser à de nouvelles stratégies pour diminuer la complexité des problèmes de GD. Ainsi en se basant uniquement sur les premiers choix de chaque liste, des approximations pourrait déjà être développées pour retourner un contrôle pas forcément optimal mais déjà très robuste. L'idée

suivante est alors d'ajouter les choix de manière progressive dans une approximation résolue itérativement avec une demande de plus en plus raffiner au fur et à mesure.

Nouvelle simulation à arrivées fluide Le problème de GD tel que modélisé de nos jours n'est pas seulement difficile à résoudre, il peut aussi devenir rapidement très long à évaluer par la simulation. Des scenarios d'arrivées de clients doivent être générés de façon à pouvoir estimer un revenu moyen qui approxime ainsi le revenu à espérer. Pour chaque scénario, un revenu est calculé en appliquant le contrôle de disponibilité aux arrivées discrètes consécutives. Le nombre de scénarios est quasiment toujours très important et il faut donc en générer beaucoup pour avoir une estimation précise. De plus, le nombre d'arrivées est souvent élevé et le revenu pour chaque scenario est alors long à calculer. L'évaluation du revenu à espérer est très utilisée en GD car il intervient aussi bien dans la validation des prédictions de la demande que pour certaines méthodes d'optimisation basées sur de la simulation. La rapidité de l'évaluation est donc primordiale et les méthodes actuelles par arrivées discrètes montrent donc leurs limites. Dans ce doctorat nous envisageons de travailler avec de la simulation par arrivées fluides pour évaluer le revenu à espérer. La demande est modélisée comme arrivant de manière fluide par segment plutôt qu'individuellement. Il s'agit d'une approximation du modèle à arrivées discrètes mais le biais engendré en pratique est souvent très faible. Cette approche est utilisée notamment pour la simulation des centres d'appels et a fait ses preuves en termes de temps de calcul et de qualité d'approximation du revenu à espérer.

CHAPITRE 2 REVUE DE LITTÉRATURE

Nous présentons dans cette revue de littérature les différentes recherches sur les formulations et approximations successives pour l'optimisation de la gestion de la disponibilité. Pour se faire, nous commençons par les premiers modèles à une seule ressource (Section 2.1) et allons jusqu'aux modèles actuels (Section 2.2) intégrant à la fois le réseau de ressource et un comportement d'achat pour modéliser la demande. Enfin nous portons une attention particulière à la littérature sur la simulation pour le problème la gestion de disponibilité (Section 2.3) à la fois en optimisation et en simple estimation.

2.1 Aspect Réseau

Les modèles à une seule ressource sont encore aujourd'hui utilisés dans la résolution de modèles réseau. Ils sont présentés brièvement en 2.1.1. La formulation exacte du modèle réseau est ensuite donnée en 2.1.2 et les approximations permettant de le résoudre pour des problèmes de grande taille sont finalement décrites en 2.1.3. Nous nous reportons au livre Talluri et van Ryzin (2004b) et l'article McGill et van Ryzin (1999) pour une revue de littérature plus complète sur ces premiers modèles.

2.1.1 Modèles à tronçon unique

Les premières recherches sur le contrôle de disponibilité s'intéressent au modèle à un seul tronçon noté l et de capacité c_l. La demande est modélisée de façon statique stochastique avec ordre d'arrivées par revenu croissant et sans comportement d'achat. Pour deux produits 1 et 2 de revenus $r_1 > r_2$, la règle de Littlewood (1972) à l'équation Littlewood établie la quantité maximale à vendre du produit le moins cher b^*_{2l} afin d'espérer le revenu optimal.

$$r_2 > r_1 \cdot P(D_1 > c_l - b^*_{2l}) \qquad \text{(Littlewood)}$$

r_j	Revenu du produit j
D_j	Demande pour le produit j
c_l	Capacité du tronçon l
b_{ji}	Quantité maximale à vendre du produit j sur le tronçon i

La partie droite de l'équation Littlewood correspond en fait au revenu marginal espéré par la vente de $c_l - b^*_{2l}$ produits 1 de plus haut revenu. Cette notion sera ensuite utilisée par les

heuristiques EMSR-a de Belobaba (1987a) et EMSR-b de Belobaba (1992) pour résoudre le problème avec n produits.

Ce problème à n produits est formulé un peu plus tard par Brumelle et McGill (1993) et Robinson (1995) en un programme dynamique dont les états sont les produits ordonnés par revenu croissant. Ils le résolvent de manière optimale par chaînage arrière et par une condition d'optimalité basée sur les coûts d'opportunité que nous expliquons dans la partie réseau.

En supprimant l'hypothèse d'ordre d'arrivées par revenu croissant, les chercheurs passent à une modélisation dynamique stochastique de la demande. Le problème est alors formulé en programmation dynamique en considérant les états comme des instants d'arrivées d'au maximum un client. La condition d'optimalité est trouvée par Lautenbacher et Stidham (1999). Nous l'expliquons dans la partie 2.1.2 réseau suivante.

2.1.2 Formulation exacte du modèle réseau (multiples tronçons)

Les recherches se tournent rapidement vers l'optimisation du réseau complet de tronçons notamment avec Glover *et al.* (1982) ou Smith et Penn (1988) pour la compagnie *Frontlier Airlines*. En tenant compte des interactions entre les tronçons partagés par plusieurs produits, Williamson (1992) ou Belobaba (2001) notent une amélioration de revenu par rapport aux modèles à unique tronçon. La modélisation réseau occupe depuis la majeure partie des recherches.

Gallego et van Ryzin (1997) proposent un programme dynamique du problème de contrôle de disponibilité de réseau pour une modélisation dynamique stochastique d'une demande sans comportement d'achat avec arrivées individuelles pour un produit. Ils formulent alors l'équation DP de Belleman et trouvent sa condition d'optimalité .

$$R_t(c) = \mathbb{E}\left[\max_{u \in U(c,j_t)} r_{j_t} u + R_{t+1}(c - uA_{j_t})\right] \qquad \text{(DP)}$$

c	Vecteur de capacité des m tronçons
$R_t(c)$	Revenu entre t et le départ pour la capacité c
j_t	Produit j arrivant éventuellement à t
r_j	Revenu du produit j
$u = u(c,t,j_t)$	Décision de vendre ou non le produit j_t à t et pour c
$U(c,j)$	Domaine $\{0,1\}$ si $c - A_{j_t} \geq 0$ et $\{0\}$ sinon
A_j	Vecteur de consommation des tronçons par le produit j

Ils trouvent aussi la condition d'optimalité DP-u* de DP en se basant sur le coût d'opportunité $\Delta_t^j(c)$ qui correspond à la perte de valeur du réseau suite à la vente à l'instant t et pour

la capacité c du produit j et donc à la perte de la capacité A_{j_t}.

$$\Delta_t^j(c) = R_t(c) - R_t(c - A_j) \qquad (\Delta_t^j(c))$$

$$u = u(c, t, j_t) = \begin{cases} 1 & si\ r_{j_t} \geq \Delta_t^j(c)\ et\ A_{j_t} \leq c \\ 0 & sinon \end{cases} \qquad \text{(DP-u*)}$$

La résolution optimale de DP est possible en théorie, mais impossible en pratique à cause du nombre d'états trop important des réseaux réels. Nous décrivons dans la partie suivante les approximations de ce programme dynamique.

2.1.3 Approximations pour le modèle réseau (multiples tronçons)

L'approximation la plus connue est le modèle *deterministic linear program* (*DLP*) de D'sylva (1982) dont le contrôle obtenu est partitionné. La demande est alors sans comportement d'achat et donc par produit. Le processus d'arrivée est statique déterministe pour tout l'horizon de réservation. Ils maximisent alors le revenu *(1)* tout en respectant la capacité *(2)* et l'espérance de la demande *(3)*.

$$\begin{array}{lcll} R^{DLP}(c) & = & \max\limits_{x}\ r^\top x & (1) \\ s.t: & & Ax \leq c \quad (\pi) & (2) \\ & & 0 \leq x \leq \mu & (3) \end{array} \qquad \text{(DLP)}$$

$R^{DLP}(c)$	Revenu optimale par le DLP pour c
c	Vecteur de capacité des m tronçons
r	Vecteur de revenu des n produits
x	Vecteur de limite de réservations des n produits
A	Matrice d'incidence $m \times n$ tronçon/produit
μ	Vecteur d'espérance de demande des n produits

Talluri et van Ryzin (1999) fondent leur *randomized linear program (RLP)* sur le *DLP*. Ils le résolvent pour plusieurs vecteurs d'espérance de demande (μ dans le DLP) générés aléatoirement et moyennent les résultats de façon à prendre en compte l'aspect stochastique. Ils améliorent ainsi les résultats du *DLP,* mais augmentent le temps de résolution . Le *probabilistic nonlinear program (PNLP)* de Ciancimino *et al.* (1999) est la formulation stochastique du *DLP* et permet aussi de considérer cet aspect. Le *PNLP* donne des résultats mitigés, car l'utilisation du contrôle partitionné dans un programme stochastique est sub optimal, car trop rigide d'après de Boer *et al.* (2002). L'aspect dynamique absent des approximations précédentes peut être intégré via des ré optimisations de ceux-ci au cours de la période

de réservation. Cooper (2002) montre qu'elles doivent se faire adéquatement sous peine de diminuer le revenu.

DeMiguel et Mishra (2006) et Bertsimas et de Boer (2005b) formulent le problème par un modèle stochastique multi étapes. La modélisation de la demande est alors statique stochastique par période sans comportement d'achat. Le manque de convexité rend souvent la résolution difficile. Chen et de Mello (2009) ré optimisent alors plusieurs fois un modèle deux-étapes convexes pour approximer le modèle multi étapes. Ils obtiennent de bons résultats sur des petits réseaux par rapport au *DLP*.

Les méthodes de décomposition par tronçon comme le *displacement-adjusted virtual nesting (DAVN)* de Smith et Penn (1988) utilisent les variables duales des contraintes de capacité des modèles précédents (π pour le DLP) pour approximer le vecteur de prix de l'offre. Le revenu de chaque produit est alors réparti sur chacun de ses tronçons grâce à ces prix de l'offre. Une hiérarchie virtuelle (*virtual nesting* en Anglais) peut alors être établie sur chaque tronçon où des méthodes de modèles à unique tronçon comme l'heuristique *EMSR-b* sont utilisées pour trouver un contrôle hiérarchisé.

Les méthodes de décomposition par *programmation dynamique* se basent sur le même principe que le *DAVN,* mais décomposent le programme dynamique donné en DP suivant chaque tronçon. Elles somment alors le résultat des programmes dynamiques de tous les tronçons pour avoir celui du réseau et ainsi pouvoir décider à tout moment si il faut accepter ou refuser une requête.

Les méthodes de prix de l'offre dont celle de Simpson (1989) utilisent comme précédemment les variables duales des contraintes de capacité des approximations pour déterminer les prix de l'offre, mais s'en servent directement comme contrôle. Ces méthodes sont un peu plus précises que celles de décomposition à condition d'être recalculées dès qu'un prix de l'offre n'est plus à jour.

Les méthodes de coût d'opportunité de Bertsimas et Popescu (2003) ou Topaloglu (2008) calculent celui-ci grâce à deux résolutions des approximations précédentes, une pour $R_t(c)$ et l'autre pour $R_t(c - A_j)$. Une requête est alors acceptée si le revenu proposé est supérieur au coût d'opportunité : $r_j \geq \Delta_t^j(c)$. Ce coût d'opportunité doit être calculé à chaque requête ou être sauvegardé à l'avance dans des tables.

La résolution des modèles de réseau est plus difficile que celle des modèles à unique tronçon, mais une simulation réaliste des contrôles obtenus montre une importante amélioration du revenu. Le comportement d'achat est intégré plus récemment au problème de contrôle de

disponibilité. Nous étudions ces nouveaux modèles dans la partie 2.2 suivante.

2.2 Aspect comportemental

Jusqu'à maintenant les modèles ne prenaient pas en compte le comportement d'achat et considéraient une demande par produit indépendante de la disponibilité des produits. En réalité, une partie des clients achète un produit plus cher si leur premier choix moins cher est indisponible (*buy up*). Réciproquement, une partie des clients capables d'acheter un produit cher achète le produit disponible le moins cher (*buy down*). Les clients sont aussi connus pour se diriger vers d'autres itinéraires ou dates de départ si leur premier choix n'est pas disponible (*buy accross*). Ces phénomènes sont aujourd'hui amplifiés par la démocratisation du transport aérien où les nouvelles compagnies à bas coût rendent la segmentation du marché de plus en plus difficile. La prise en compte du comportement d'achat parait alors indispensable pour se protéger voir profiter de ces phénomènes malgré des modèles encore plus difficiles à résoudre.

Les différentes modélisations du comportement d'achat utilisées dans les modèles de contrôle de disponibilité sont décrites en 2.2.1. La formulation exacte du modèle de réseau avec comportement d'achat est ensuite donnée en 2.2.2 avant d'en étudier les approximations en 2.2.3.

2.2.1 Modélisations du comportement d'achat

Multinomial logit (MNL)

La modélisation du comportement d'achat la plus étudiée jusqu'à présent est le *multinomial logit (MNL)* de Ben-Akiva et Lerman (1985) basée sur de la régression logistique multinomial. Chaque client d'un segment s attribue un poids d'achat v_{js} pour chaque produit j mesurant ainsi son intérêt relatif pour celui-ci. Un poids nul correspond à un non-intérêt donc un non-achat du produit. Les probabilités d'arrivées sont par segment et sont notées λ_s.

Exemple : Le segment H de la table 2.1 est intéressé par l'achat des produits 1, 2 avec respectivement des poids de 5 et 10. Le dernier poids de 2 (v_{0s}) correspond au fait de ne rien acheter.

Tableau 2.1 Exemple de modélisation de comportement d'achat de type multinomial logit *(MNL)* pour deux segments (H et L) et quatre produits

Segment s	λ_s	$[v_{1s}, v_{2s}, v_{3s}, v_{4s}, v_{0s}]$
H	λ_H	$[5, 10, 0, 0, 2]$
L	λ_L	$[0, 0, 5, 1, 5]$

La probabilité d'achat $P_j(S)$ d'un produit j change alors suivant l'ensemble S de produits

offerts. On la calcule grâce à l'équation 2.1 où $|s|$ est le nombre de segments.

$$P_j(S) = \sum_{s=1}^{|s|} \lambda_s \frac{v_{js}}{v_{0s} + \sum_{k \in S} v_{ks}} \tag{2.1}$$

Liste de préférence (PL)

Dans la modélisation par liste de préférence *(PL)*, chaque segment possède une liste de préférence pl_s qui correspond à l'ordre dans lequel le client va acheter les produits dont il est intéressé.

Exemple : Le segment L du Tableau 2.2 achète d'abord le produit p_1, puis p_2 si p_1 n'est pas disponible et enfin il quittera si ni p_1 ni p_2 ne sont disponibles.

Tableau 2.2 Exemple de modélisation de comportement d'achat par liste de préférence *(PL)*

Segment s	λ_s	*Liste de préférence*
H	λ_H	$p_2 \rightarrow p_3 \rightarrow p_4 \rightarrow quitt$
L	λ_L	$p_1 \rightarrow p_2 \rightarrow quitt$

La probabilité d'achat $P_j(S)$ d'un produit j change alors suivant l'ensemble S de produits offerts. On la calcule grâce à l'équation 2.2 où $|s|$ est le nombre de segments et $pl_s(j)$ est la hiérarchie de j dans la liste de préférence pl_s (1 est le produit de plus haute hiérarchie).

$$P_j(S) = \sum_{s=1}^{|s|} \lambda_s \mathbb{I}\left\{\forall k \in pl_s \cap S : pl_s(j) < pl_s(k)\right\} \tag{2.2}$$

Chaneton et Vulcano (2011) proposent une méthode empirique pour passer sans équivalence de la modélisation *MNL* vers la *PL* en utilisant l'ordre croissant de poids comme ordre dans la liste de préférence.

Ordre de préférence (PO)

Dans la modélisation par ordre de préférence, une probabilité de quitter le système de réservation est ajoutée entre chaque produit successif de chaque liste de préférence de segment. La modélisation par ordre de préférence permet d'agréger plusieurs listes de préférence comme le montrent Hosseinalifam *et al.* (2014) (deuxième article).

Exemple : Le segment L du Tableau 2.3 achète d'abord le produit p_1. Puis si p_1 n'est pas disponible, il achètera p_2 avec probabilité $0,4$ ou quittera avec une probabilité $0,6$. Enfin il

quittera si ni p_1 ni p_2 ne sont disponibles.

Tableau 2.3 Exemple de modélisation de comportement d'achat par ordre de préférence *(PO)*

Segment s	λ_s	*Liste de préférence*	*Probabilité de transition*
H	λ_H	$p_2 \to p_3 \to p_4 \to quitt$	0,8 - 0,7 - 1
L	λ_L	$p_1 \to p_2 \to quitt$	0,4 - 1

La probabilité d'achat $P_j(S)$ d'un produit j en fonction de l'ensemble S de produits offerts se calcule de la même manière que pour la modélisation par liste de préférence en multipliant en plus par les différentes probabilités de transition.

2.2.2 Formulation exacte

L'ajout du comportement d'achat au modèle de réseau en 2.1.2 précédent est illustré par la formulation CDP. Les arrivées de produits j_t sont alors remplacées par des arrivées éventuelles de segments s_t qui choisissent un produit $j(s_t, J_t)$ en fonction de leur comportement d'achat et des produits J_t disponibles à t.

$$R_t(c) = \mathbb{E}\left[\max_{u \in U(c, j(s_t, J_t))} \{r_{j(s_t,J_t)} u + R_{t+1}(c - uA_{j(s_t,J_t)})\}\right] \qquad \text{(CDP)}$$

c	Vecteur de capacité des m tronçons
$R_t(c)$	Revenu entre t et le départ pour la capacité c
r_j	Revenu du produit j
s_t	Segment arrivant éventuellement à t
J_t	Produit(s) disponible(s) à t.
$j(s_t, J_t)$	Produit $j \in J_t$ choisit par s_t à t
$u = u(c, t, j(s_t, J_t))$	Décision de vendre ou non le produit $j(s_t, J_t)$ à t et pour c
$U(c, j)$	Domaine $\{0, 1\}$ si $c - A_j \geq 0$ et $\{0\}$ sinon
A_j	Vecteur de consommation des tronçons par le produit j

Quelque soit la modélisation du comportement d'achat, les conditions d'optimalité en DP-u* demeurent valides. Néanmoins l'ajout du comportement d'achat rend la résolution encore plus difficile. Le modèle exact est donc approximé par des modèles que nous étudions dans la partie suivante.

2.2.3 Approximations avec comportement d'achat

Le *choice based determinitic linear program (CDLP)* de Gallego *et al.* (2004) est le modèle statique déterministe de réseau avec comportement d'achat. Il est statique déterministe et peut être comparé à un DLP avec comportement d'achat. À chaque instant, on peut mettre

en vente un ensemble S d'un ou plusieurs produits. On veut alors maximiser le revenu en trouvant les durées de mise en vente optimales t_S *(4)* pour chaque ensemble S *(1)* tout en respectant la capacité *(2)* et en ne dépassant pas la durée de la période de réservation *(3)*. Les espérances de revenu R_S et de consommation des tronçons Q_S de chaque ensemble S par unité de temps sont calculées grâce aux probabilités d'achat. Le *CDLP* peut donc être utilisé avec toutes les modélisations vues en 2.2.1.

Le nombre d'ensembles possibles est exponentiel (2^n-1). Pour des réseaux de taille réaliste, le *CDLP* doit alors se résoudre par génération de colonnes dont le sous-problème est NP-difficile dans le cas général d'après Bront *et al.* (2009). Dans le cas où les segments sont disjoints (un seul segment par produit) et pour la modélisation *MNL*, Liu et van Ryzin (2008) introduisent la notion d'ensemble efficace permettant une résolution rapide du *CDLP*. Ils notent une amélioration de 5% à 10% par rapport au modèle de demande indépendante par produit sur leurs instances. Le *CDLP* fournit une solution statique avec des durées non ordonnées de mise en vente et donc non utilisable directement. Liu et van Ryzin (2008) développent, dans le cas de segments disjoints, une méthode combinant *CDLP* avec une décomposition de CDP par tronçon en utilisant des prix de l'offre statiques. Zhang et Adelman (2009) améliore cette méthode en utilisant une décomposition avec des prix de l'offre dynamiques.

$$
\begin{aligned}
R^{CDLP}(c) = \quad & \max_{t_S} \quad \sum_{S \subseteq N} \lambda R_S t_S && && \textit{(1)} \\
& s.t: \quad \sum_{S \subseteq N} \lambda Q_S t_S \leq c && && \textit{(2)} \\
& \qquad \sum_{S \subseteq N} t_S \leq T && && \textit{(3)} \\
& \qquad t_S \geq 0 && \forall S \subseteq N && \textit{(4)}
\end{aligned}
\tag{CDLP}
$$

c	Vecteur de capacité des m tronçons
$R^{CDLP}(c)$	Revenu généré par le CDLP pour c
S	Ensemble d'un ou plusieurs produits
N	Ensemble des 2^n ensembles de produits
t_S	Durée de mise en vente de l'ensemble S
λ	Probabilité d'arriver à chaque instant
T	Durée de la période de réservation
$R_S = r^\top P(S)$	Revenu espéré par unité de temps de l'ensemble S
$Q_S = AP(S)$	Vecteur de consommation espéré des *tronçons* par unité de temps de l'ensemble S
$P(S)$	Vecteur de probabilité d'achat des n produits pour l'ensemble S
r	Vecteur de revenu des n produits
A	Matrice d'incidence $m \times n$ tronçon/produit

Le CDP est aussi approximé par Kunnumkal et Topaloglu (2008), Kunnumkal et Topaloglu (2010) et Meissner et Strauss (2012) via une décomposition par tronçon ou un regroupement de périodes. Chen et de Mello (2010) utilisent une modélisation de la demande par ordre

de préférence dans un modèle stochastique à recours retournant un contrôle partitionné. Les solutions obtenues sont souvent plus précises que celles du *CDLP,* mais les résolutions sont presque toujours plus compliquées et les modèles sont donc difficilement utilisables en pratique. Talluri (2010) propose le modèle de *segment-based deterministic concave-program (SDCP)* approximant le CDP par une formulation concave plus facile à résoudre que le *CDLP*. Talluri (2014) améliore un peu la précision en intégrant l'aspect stochastique avec plusieurs évaluations du *SDCP* pour différentes réalisations de la demande dans son *randomized concave program (RCP)*. L'utilisation du *MNL* dans le *SDCP* donne le *sales-based linear programming (SBLP)*. Meissner *et al.* (2013) ajoutent alors des contraintes sur les produits pour améliorer la précision de ce dernier qui devient presque aussi précis que le *CDLP*. Le temps de résolution est raisonnable tant que le nombre de chevauchements entre les ensembles de considération reste petit.

Plus récemment Hosseinalifam *et al.* (2014) proposent une formulation équivalente du *CDLP* compatible avec toutes les modélisations vues en 2.2.1 et résolue par génération de colonnes. Ils retournent alors des prix de l'offre dynamiques dans leur premier article, des durées de mises à disposition d'ensemble de produits dans leur deuxième article. Dans leur troisième article, ils rajoutent certaines règles commerciales. Pour les instances testées, les revenus trouvés sont souvent meilleurs que les autres approches vues précédemment et les temps de résolutions pour des instances de taille importante sont relativement faibles.

van Ryzin et Vulcano (2008a) avec un contrôle hiérarchisé et Chaneton et Vulcano (2011) avec des prix de l'offre utilisent une modélisation de la demande par liste de préférence dans un modèle d'optimisation basée sur la simulation. Nous expliquerons plus en détail ces modèles dans la partie suivante. Les premiers appliquent leur méthode à un réseau de compagnie aérienne et améliorent jusqu'à 10% le revenu par rapport à celui d'un *DLP/DAVN* sans comportement d'achat. Pour un réseau de trois vols parallèles, les seconds observent jusqu'à 30% d'amélioration de revenu par rapport à un *DLP/DAVN*. Ces résultats sont à relativiser, car ces méthodes considèrent aussi un aspect dynamique et stochastique absent du *DLP/DAVN*.

La prise en compte du comportement d'achat améliore les solutions en rendant le contrôle plus robuste face aux phénomènes de *buy up*, *buy down* ou *buy accros*. En contrepartie, le temps de résolution augmente significativement. L'ajout de l'aspect stochastique et dynamique à ces modèles souvent statiques déterministes semble alors être un réel défi. Nous présentons dans la partie 2.3 suivante les modèles d'optimisation basée sur la simulation qui intègrent tous ces aspects.

2.3 Optimisation basée sur la simulation

La simulation n'est pas un modèle de recherche de solution optimale comme la programmation mathématique, mais un modèle imitant le système et permettant d'évaluer n'importe quelle solution. En raison de l'absence de mécanique d'optimisation, elle évalue rapidement le résultat d'une solution, et ce même pour une modélisation complexe permettant un grand réalisme. Très tôt, des chercheurs comme Williamson (1992) l'utilisent pour évaluer des contrôles, solutions de programmes mathématiques à modélisation incomplète, dans un contexte plus réaliste. La simulation comme définie en 2.3.1 est aujourd'hui indispensable pour juger de manière réaliste de la qualité des contrôles d'un modèle ou d'une méthode de résolution de contrôle de disponibilité. Plus récemment, des techniques d'optimisation basée sur la simulation se développent comme nous le verrons dans les parties 2.3.2, 2.3.3 et 2.3.4.

2.3.1 Évaluation par simulation

Le modèle de simulation utilisé dans les recherches pour évaluer et comparer des contrôles est presque toujours dynamique stochastique à comportement d'achat. On note $\tilde{W} = \{s_t : 0 \leq t \leq T\}$ le processus aléatoire d'arrivées des segments à tout instant. Pour effectuer une évaluation par la simulation, on génère alors les arrivées W suivant la distribution de $\tilde{W}$. En partant de $t = 0$, par la formule de récurrence SIMUL et en appliquant le contrôle connu u, on arrive à déterminer les produits disponibles J_t à tout instant t. Le revenu $R(c, u, W) = R_0(c, u, W)$ propre à W peut alors être déduit.

$$R_t(c, u, W) = r_{j(s_t, J_t)} u + R_{t+1}(c - uA_{j(s_t, J_t)}) \qquad \text{(SIMUL)}$$

c	vecteur de capacité des m tronçons
W	arrivées générées s_t entre 0 et T avec $W \sim \tilde{W}$
$R_t(c, u, W)$	revenu entre t et le départ pour c pour W avec le contrôle u
r_j	revenu du produit j
s_t	segment arrivant éventuellement à t
J_t	produit(s) disponible(s) à t.
$j(s_t, J_t)$	produit $j \in J_t$ choisit par s_t à t
$u = u(c, t, j(s_t, J_t))$	décision de vendre ou non le produit $j(s_t, J_t)$ à t et pour c
A_j	vecteur de consommation des tronçons par le produit j

Le revenu obtenu est propre au W généré. Pour avoir un estimé de l'espérance du revenu $R(c, u, \tilde{W}) = \mathbb{E}[R(c, u, W)]$, on moyenne alors n évaluations à la manière de la simulation Monte-Carlo comme illustrée à l'équation MonteCarlo. On pourra alors en déduire une valeur

moyenne ainsi qu'un intervalle de confiance.

$$R(c,u,\tilde{W}) = \mathbb{E}[R(c,u,W)] \approx \overline{R}(c,u,\tilde{W},n) = \frac{1}{n}\sum_{k=1}^{n} R(c,W_k) \qquad \text{(MonteCarlo)}$$

La variance permettant d'établir l'intervalle de confiance est alors estimée empiriquement par l'estimateur 2.3.

$$Var[R(c,u,W)] \approx \frac{1}{n-1}\sum_{k=1}^{n}(R(c,u,W_k) - \overline{R}(c,u,\tilde{W},n))^2 \qquad (2.3)$$

Ce revenu est ainsi calculé dans un contexte réaliste puisque les aspects de dynamisme, l'aspect stochastique et comportement d'achat sont considérés. La simulation permet donc d'évaluer tout contrôle de façon très réaliste.

2.3.2 Optimisation basée sur la simulation

Malgré son nom, l'optimisation basée sur la simulation combine en fait un programme mathématique (étape 1) et méthode de convergence sur simulation (étape 2) comme illustrée à la figure 2.1. L'objectif final est de trouver un bon contrôle de la disponibilité dans un contexte réaliste donc pour une modélisation précise : dynamique, stochastique et avec comportement d'achat.

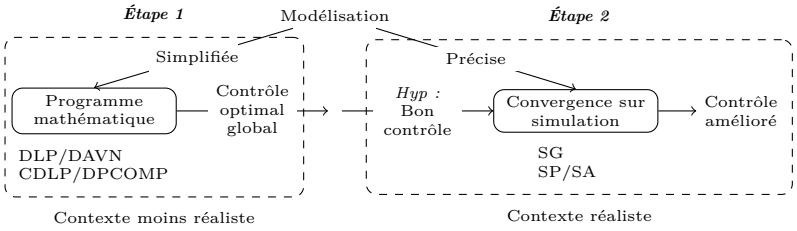

Figure 2.1 Techniques d'optimisation basée sur de la simulation

L'étape 1 consiste à résoudre un programme mathématique de modélisation à contexte un peu moins réaliste de façon à trouver un contrôle optimal en un temps raisonnable. Ce contrôle noté u_0 ne change pas suivant le contexte, mais est uniquement optimal dans le contexte moins réaliste. La méthode suppose toutefois que celui-ci appartient à une région de bons contrôles pour le contexte réaliste et devrait donc être une bonne solution de départ dans ce nouveau contexte. Le compromis de modélisation pour le programme mathématique est souvent fait sur l'aspect stochastique ou sur le comportement d'achat. Ainsi la plupart des recherches utilisent un DLP avec DAVN.

Dans ***L'étape 2***, on utilise alors le modèle de simulation qui correspond à un contexte réaliste. La simple évaluation par la simulation du contrôle u_0 dans ce contexte plus réaliste peut entraîner un revenu simulé $R(c, u_0)$ différent. Le but est alors d'améliorer ce contrôle initial en utilisant la simulation et son contexte réaliste grâce à une méthode de convergence basée sur la simulation. Nous décrivons les grandes étapes de ce type de méthodes ci-dessous :

1. On part du contrôle initial $u \leftarrow u_0$
2. On calcul le gradient du revenu par rapport à u : $\nabla_u R(c, u, \tilde{W})$
3. On modifie u en fonction du gradient : $u \leftarrow Proj_\Theta(u + \gamma \nabla_u R(c, u, \tilde{W}))$ avec γ le pas et Θ l'ensemble des contrôles admissibles.
4. Si le critère d'arrêt n'est pas atteint (norme du gradient, nombre d'itérations, amélioration de revenu ...) on retourne en 2 sinon STOP.

Ces méthodes de convergence sont basées sur la simulation, car le gradient $\nabla_u R(c, u, \tilde{W})$ et le revenu $R(c, u, \tilde{W})$ sont calculés via évaluation par simulation. Suivant les paramètres de la méthode de convergence sur simulation, le contrôle amélioré peut parfois être prouvé comme optimal localement. Les méthodes d'optimisation basées sur la simulation se divisent en deux groupes suivants que le calcul du gradient se fait indépendamment du modèle de simulation comme en 2.3.3 ou non comme en 2.3.4.

2.3.3 Calcul du gradient sans utilisation du modèle

Lorsque le modèle de simulation n'est pas utilisé par la technique de convergence, on parle d'optimisation de boite noire. La méthode de convergence utilise alors uniquement le revenu d'évaluation par simulation de plusieurs contrôles pour converger. Bertsimas et de Boer (2005a) se servent alors de différences finies classiques pour calculer chacune des coordonnées du gradient. Gosavi *et al.* (2005) utilisent une méthode de perturbations simultanées basée sur la différence entre des évaluations dont le contrôle a été légèrement et aléatoirement perturbé pour calculer le gradient. Cette méthode semble plus efficace que la première en terme de temps de calcul. Les différences finies demandent d'effectuer un très grand nombre d'évaluations. D'autant plus que cette opération est répétée à chaque itération de l'algorithme de gradient stochastique. Bertsimas et de Boer (2005a) coupent alors l'horizon de réservation en deux et se basent sur une estimation du revenu pour la période plus lointaine qui est mise à jour moins fréquemment contrairement à la partie plus proche.

Une fois le gradient déterminé, la méthode de descente de gradient stochastique expliquée en 2.3.2 est utilisée pour tenter d'améliorer le revenu. Cette méthode est convergente sous certaines conditions sur le pas et la fonction objectif. Bertsimas et de Boer (2005a) et Gosavi

et al. (2005) montrent tous les deux que ces conditions ne sont pas remplies dans leurs modèles. Néanmoins dans ces deux recherches, des améliorations sont toujours constatées par rapport au contrôle de première étape issue d'un *DLP/DAVN*. Elles sont par exemple de l'ordre de 1 à 2% pour un réseau de quinze villes et un centre (*hub*) dans Bertsimas et de Boer (2005a).

Gosavi *et al.* (2005) étudient également la méthode de recuit simulé qui est une méta heuristique de recherche globale. Les résultats sont légèrement moins bons que la méthode de perturbation simultanée, mais les auteurs ne fournissent pas de temps de calcul qui permettraient de les comparer complètement.

2.3.4 Calcul du gradient avec utilisation du modèle

Le deuxième type de techniques utilise des propriétés du modèle pour calculer le gradient. Ce calcul n'est souvent plus valide lorsque le modèle est modifié. Une expression exacte du gradient est alors obtenue suivant le modèle retenu. Pour les modèles avec comportement d'achat par liste de préférence, van Ryzin et Vulcano (2008a) expriment ce gradient pour un contrôle hiérarchisé et Chaneton et Vulcano (2011) avec des prix de l'offre. Pour se faire, ils assument la demande et la capacité comme continue et se servent d'une petite variation aléatoire de la capacité de chaque tronçon pour rendre le problème différentiable. van Ryzin et Vulcano (2008b) et Topaloglu (2008) font de même pour un modèle sans comportement d'achat. Dans Topaloglu (2008) et van Ryzin et Vulcano (2008a) la fonction de décision $u(x, t, j)$ est aussi assumée comme continue rendant ainsi les réservations partielles possibles. Dans tous les cas, l'expression est dérivée de la formule SIMUL.

La méthode de convergence utilisée est encore celle du gradient stochastique présentée en 2.3.2. Les hypothèses faîtes pour le calcul du gradient permettent alors de prouver au moins une convergence avec probabilité vers un optimum local. Sauf pour Chaneton et Vulcano (2011) où la convergence avec probabilité est seulement vers un point stationnaire. Les contrôles initiaux sont calculés par un modèle de première étape de type *DLP/DAVN* pour le contrôle hiérarchisé et avec les valeurs duales du *DLP* pour le contrôle par prix de l'offre.

Les contrôles initiaux sont quasiment toujours améliorés dans les instances testées. Topaloglu (2008) surpasse jusqu'à 3% les revenus des méthodes plus classiques de prix de l'offre basées sur les valeurs duales de *DLP*, de *RLP* et sur des coûts d'opportunité calculés *DLP*. van Ryzin et Vulcano (2008a) dépassent de 10% les revenus de méthodes classiques pour une instance d'un réseau d'une compagnie aérienne même si le résultat est à relativiser, car sa méthode intègre du comportement d'achat que n'ont pas les modèles auquel il se compare.

van Ryzin et Vulcano (2008b) montrent que leur méthode est plus rapide que celle de Bertsimas et de Boer (2005a) de type boite noire. De manière générale les quatre articles concluent sur un temps de calcul suffisamment raisonnable pour que leur méthode soit appliquée en pratique. van Ryzin et Vulcano (2008b) voudraient tout de même étudier les méthodes de réduction de la variance pour réduire le temps de calcul. van Ryzin et Vulcano (2008a) proposent d'utiliser deux processeurs de façon à accélérer la résolution avec du calcul en parallèle.

CHAPITRE 3 ORGANISATION DE LA THÈSE

Cette thèse porte sur la résolution et l'affinement de la modélisation du problème de gestion de la disponibilité (PGD) lorsque plusieurs ressources sont considérées et que la demande suit un comportement d'achat. Comme présenté au chapitre 1, ce problème appartient au domaine de la gestion de revenu (GR) plus usuellement dénommé *revenue management*. Au chapitre 2 précédent, nous étudions la littérature relative à ce problème et présentons l'état de l'art des modélisations et de leur résolution ou approximation. Conscients de ces différentes approches, nous proposons deux paradigmes : le premier aborde la politique de disponibilité par des temps de fermeture par produit alors que le deuxième considère les arrivées comme fluides pour la simulation. Ces deux paradigmes permettent d'accélérer grandement la résolution du problème d'optimisation et sa simulation tout en maintenant une modélisation très réaliste.

Résolution par temps de fermeture lorsque le comportement d'achat est non-paramétrique. Au chapitre 4, nous proposons une approximation pour le problème de gestion de la disponibilité pour un comportement de type non paramétrique. Nous introduisons une nouvelle politique de disponibilité par temps de fermeture de vente pour chaque produit. Celle-ci nous permet de proposer une approximation non linéaire particulièrement adaptée au non-paramétrique. Cette politique de disponibilité empêche naturellement la réouverture à la vente des produits. Dans ce cas particulier de non réouverture, nous montrons que notre approximation est une borne supérieure pour la formulation exacte du problème et est asymptotiquement optimale. Nous montrons que notre politique par temps de fermeture est équivalente à une politique de disponibilité utilisée par les principales approximations existantes et autorisant la réouverture. Nos solutions peuvent donc être utilisées comme solution initiales pour ces approximations et de cette façon, notre politique peut être réouverte. Notre approximation est linéarisée grâce à des variables binaires représentant une hiérarchie de produit qui a un réel sens pratique. Cela nous permet de proposer de bonnes solutions initiales très facilement. Nous menons des expériences numériques sur trois instances représentant chacune un type et une taille d'instance bien particulière. Notre approximation retourne beaucoup plus rapidement une politique de disponibilité générant un revenu espéré légèrement supérieur aux approches existantes. Ils mettent aussi en évidence l'accélération des approximations existantes lorsque notre approche est utilisée comme solution de départ.

Généralisation de la formulation par temps de fermeture à tout type de comportement d'achat. Au chapitre 5, nous représentons tout comportement d'achat en chemins

d'achats dont la structure est non paramétrique. Ces différents chemins d'achats forment un arbre que nous exploitons avec l'approximation du chapitre 4 pour proposer une approximation acceptant toute modélisation de la demande. Celle-ci partage les mêmes caractéristiques : temps de fermeture, réouverture et nous arrivons à étendre les résultats théoriques précédents sur la borne supérieure de la formulation exacte et sur l'optimalité asymptotique. Cependant, les modèles paramétriques engendrent un arbre de demande immense qui rend notre approximation insoluble. Pour pallier cela, nous présentons une méthode de résolution itérative basée sur une construction progressive de l'arbre de demande par un ajout successif de chemins d'achats. Nous proposons plusieurs heuristiques comme stratégie d'ajout de chemin. Des expériences numériques sont menées sur les mêmes instances que précédemment sauf que le comportement d'achat est paramétrique cette fois-ci. Les résultats montrent surtout l'excellent performance de la méthode itérative de résolution qui converge vers une bonne solution beaucoup plus rapidement que les autres approches.

Modélisation par arrivées fluides pour la simulation. Au chapitre 6, nous développons la simulation pour la gestion de disponibilité. Nous proposons ainsi un nouvel estimateur à arrivées fluides permettant de déterminer le revenu espéré. Notre modèle agrège les différentes arrivées par segment, et ce pour toute la période de réservation. Il ne subit donc pas l'aléatoire d'un ordre d'arrivées. Il utilise uniquement une évaluation et est invariant alors que l'estimateur traditionnel à arrivées discrètes ne peut réduire sa variance qu'en augmentant le nombre d'évaluations. En contrepartie, nous montrons que notre estimateur présente un biais, contrairement à l'estimateur traditionnel. Nous montrons que ce biais peut être arbitrairement grand même si cela est dû à des aspects minimes pour des instances pratiques. Nous testons alors des méthodes d'optimisation basées sur la simulation et concluons de l'importance du point de départ. Celui-ci est fourni par les approximations vues précédemment et nous montrons que notre estimateur est équivalent à la plupart des approximations pour le contrôle de la disponibilité. En conséquence, nous évoquons quelques possibilités de notre estimateur pour appuyer l'optimisation. Les résultats numériques sur des instances de grandes tailles témoignent de la nette supériorité de notre estimateur en termes de temps de calcul. Nous constations aussi peu de biais pour toutes les instances étudiées.

Enfin, nous proposons au chapitre 7 une discussion générale sur les contributions et les limites de ce travail. Une conclusion est finalement donnée au chapitre 8.

CHAPITRE 4 ARTICLE 1: PRODUCT-CLOSING APPROXIMATION FOR NONPARAMETRIC CHOICE NETWORK REVENUE MANAGEMENT

Le texte de ce chapitre est celui de l'article *Product-Closing Approximation for Nonparametric Choice Network Revenue Management* soumis à *European Journal of Operational Research*. Auteurs : Thibault Barbier, Miguel F. Anjos, Fabien Cirinei et Gilles Savard.

Abstract

Most recent research in network revenue management incorporates choice behavior that models the customers' buying logic. These models are consequently more complex to solve, but they return a more robust policy that usually generates better expected revenue than an independent-demand model. Choice network revenue management has an exact dynamic programming formulation that rapidly becomes intractable. Approximations have been developed, and many of them are based on the multinomial logit demand model. However, this parametric model has the property known as the independence of irrelevant alternatives and is often replaced in practice by a nonparametric model. We propose a new approximation called the product closing program that is specifically designed for nonparametric demand. Numerical experiments show that our approach quickly returns expected revenues that are slightly better than those of other approximations, especially for large instances. Our approximation can also supply a good initial solution for other approaches.

4.1 Introduction

In 1978, when the US airline market was deregulated, airlines lost their quasi-monopolistic status, moving to a competitive market. They were forced to improve efficiency, in terms of both operation productivity and sales profitability. Operation productivity optimization aims to improve the scheduling, maintenance, and assignment of limited resources. Sales profitability optimization is a type of revenue management: it aims to maximize the revenue obtained from perishable resources. These issues are considered separately because the subproblems are tractable whereas the overall problem is too complex. Today, scheduling and revenue management have many applications: airlines, rental car companies, and hotels.

We focus on the revenue management problem for which perishable resources are sold through different products to customers during a reservation period. Selling a low price product early consumes a resource that could perhaps have been sold later at a better price. However,

holding on to resources for future sales fails to satisfy the current demand. The challenge is thus to control the availability of products, also called availability policy, over the reservation period to maximize revenue. The resources, products and demand are known and fixed. This problem is not to be confused with pricing or assortment which are different revenue management problems even if there are similarities. The revenue management in this article refers to the problem of availability policy.

Research has shown that it is better to optimize the network formed by the resources rather than each resource individually, but this leads to larger problems. The latest trend in revenue management is the implementation of choice behavior instead of an assumption of independent demand. The problem is more complex, but the solutions are more accurate and robust. This version of revenue management is referred to as the choice network revenue management problem (CNRM). It was first introduced by Gallego *et al.* (2004), and an exact dynamic programming (DP) formulation was given by Talluri et van Ryzin (2004a).

However, the DP rapidly becomes intractable because of the number of states. Researchers have therefore proposed various approximations, returning solutions that are either dynamic or static. The quality of the approximation can be measured by the expected revenue and the solution time. The most popular approximations are the choice deterministic linear program (CDLP) proposed by Liu et van Ryzin (2008), which is static, and DP decomposition by resources, which is dynamic. For large instances and especially because of the choice behavior, these approximations are large and difficult to solve. The multinomial logit (MNL) model as explained in Ben-Akiva et Lerman (1985) and Hanson et Martin (1996), which is widely used in the marketing and economics literature, is often used for the choice behavior. Many methods such as column generation and heuristics have been developed for this model because its structure is well-accommodated for estimation and CNRM approximations.s

However, the MNL model has an important drawback known as the independence of irrelevant alternatives (IIA) as detailed in Ratliff *et al.* (2008). IIA causes improbable substitutions when products share similar characteristics. Unfortunately perfect substitutes, such as the red/blue bus example of Ben-Akiva et Lerman (1985), often occur in revenue management. Moreover, the data available for forecasting may better fit another demand model. We focus on the preference list (PL) model which is a nonparametric alternative to the multinomial logit. In the former, customers choose from an ordered list of ranked products. A probability of transition is specified between each pair of products. Our work is motivated by the fact that most recent researches on choice behavior models have focused on PL estimation as in Farias *et al.* (2013), van Ryzin et Vulcano (2015) and more recently van Ryzin et Vulcano (2017). On the other hand, there has been limited research into PL or nonparametric CNRM

approximations, and most of the studies are adaptations of existing MNL approaches.

Our approximation exploits the structure of the PL model and is not based on an existing approximation. By taking advantage of the logical transitions between products rather than working with sets of products as in MNL approximations, we avoid the extremely high number of product combinations. This results in a nonlinear model that can easily be linearized, and the binary variables have a practical significance that can be exploited to provide good initial solutions. The complexity of our model depends linearly on the number of products considered for each segment. Unlike many other approximations, our formulation benefits from overlapping by reusing variables when different segments share products; this reduces the complexity. We assume nonreopening: products are sold until a specified time and then never sold again. Some companies have such a strategy, and most approximations model it via additional constraints that slow the solution process. When reopening is allowed, our approximation can return a set of product durations that can serve as a good initial solution for an approximation that allows reopening.

Our experiments show promising results in comparison with other approximations. Our approximation returns an equivalent or better expected revenue in a shorter solution time for all the instances, although there is nonreopening. The results also demonstrate the time saved by using our solution as an initial solution for an approximation with reopening. We also show the limitations of some current approximations for the largest instances, to highlight the practical feasibility of our approach.

The remainder of this paper is organized as follows. In Section 4.2, we review the CNRM literature, especially with nonparametric choice behavior. In Section 4.3, we introduce the notation and give the exact formulation of CNRM. In Section 4.4, we present our approximation with preference-list choice behavior and its theoretical properties. In Section 4.5, we present practical methods for the efficient solution of our approximation. Numerical experiments and approximation benchmarks are reported in Section 4.6, and Section 4.7 provides concluding remarks.

4.2 Related literature

We refer to Talluri et van Ryzin (2004b) for reviews of the historical revenue management problem with or without the network and choice aspects. Strauss *et al.* (2018) presents the most recent researchs on the general revenue management with choice behavior. We focus on the CNRM problem and discuss only the relevant literature.

As mentioned in the Introduction, this problem has an exact DP formulation Talluri et van

Ryzin (2004a). Because it rapidly becomes intractable, approximations have been proposed in two categories: static and dynamic.

The static approximations are based on the expected demand. They therefore reduce the complexity and can tackle larger instances but ignore the demand uncertainty. The solution obtained is not updated in response to new arrivals and is hence called static. In this category, CDLP Liu et van Ryzin (2008) is the most widely used. It indicates for how long each set of products, also called an offer, must be sold over the reservation period. By empirically ordering the offers and their durations over the reservation period, we obtain a static policy by offer period. The CDLP is an upper bound on the DP solution and is asymptotically equivalent as resources and demand increase. However, it has an exponential number of columns and must be solved by column generation, which has an NP-hard subproblem, as explained by Bront *et al.* (2009) and Rusmevichientong *et al.* (2014). Liu et van Ryzin (2008) and Bront *et al.* (2009) propose exact and heuristic subproblem formulations for the MNL choice behavior.

The CDLP primal solution has to be ordered and gives a static policy. Liu et van Ryzin (2008) and Bront *et al.* (2009) use the optimal dual values to calculate the capacity marginal values in a DP decomposition by resource. Zhang et Weatherford (2017) and Erdelyi et Topaloglu (2010) are other approximations for the calculus of the network marginal values. In the same vein Kunnumkal et Topaloglu (2010) uses the revenue attributed to each resource rather than dual values. The dynamic policy obtained indicates what offer to propose as a function of the remaining time and capacities. However, this approach needs to solve an NP-hard problem for each resource, each time, and each remaining capacity and can therefore be intractable even if computed offline. Moreover, an NP-hard problem must be solved for each arrival to determine what offer to propose. This may be incompatible with current reservation systems.

Zhang et Adelman (2009), and Meissner et Strauss (2012) return a dynamic policy with an affine relaxation or a piecewise-linear formulation, but they consider only disjoint segments, which are rare in practice. To overcome the static aspect of the CDLP, Kunnumkal et Topaloglu (2011) base their optimization on random samples of demand while Jasin et Kumar (2012) re-solve the CDLP several times over the horizon period.

Talluri (2010) proposed the segment-based deterministic concave program (SDCP), considered as a CDLP decomposition by segment. It is more tractable if the consideration sets are not too large, but it also provides a weaker upper bound than CDLP unless the segments do not overlap, which is rare in practice. To tighten the SDCP formulation with choice behavior, Meissner *et al.* (2013) add valid constraints referred to as product cuts, Talluri (2014)

uses a random customer-arrival stream and Strauss et Talluri (2017) proves an equivalence with CDLP when the intersection of segment consideration forms a tree or consideration sets are nested. However, even with the extra constraints, no primal policy is returned and the dynamic decomposition is the principal solution. The sales-based linear program (SBLP) introduced by Gallego *et al.* (2011) and developed further by Gallego *et al.* (2014) and Talluri (2014) is a compact formulation of the SDCP under the MNL choice behavior.

Apart Hosseinalifam *et al.* (2016) with a CDLP subproblem and Chaneton et Vulcano (2011) with a stochastic gradient descent, there is currently not many researches in the CNRM policy problem for nonparametric choice behavior. For the assortment problem, Jagabathula et Rusmevichientong (2017) reaches the same conclusions and propose a complete nonparametric approach. However the recent advances on nonparametric choice behavior evaluation, such as Farias *et al.* (2013), van Ryzin et Vulcano (2015) and van Ryzin et Vulcano (2017), open the way to new researches and approaches for the policy, assortment and pricing CNRM.

4.3 Model

We start by introducing the notation for the CNRM problem. A resource $i \in I$ has a capacity c_i. There is $m = |I|$ resources. A product $j \in J$ is defined by a fare r_j and consumed one or more resources. There is $n = |J|$ resources An offer $S \subseteq J$ denotes a set of distinct products. The incidence matrix $A = [a_{ij}]_{i \in I, j \in J}$ has a_{ij} equal to 1 if resource i is used by product j and 0 otherwise.

A_j refers to the column of product j in the incidence matrix. Customers arrive during a reservation period, indexed by t, starting at $t = 0$ and finishing at $t = T$ when the resources perish. A segment $l \in L$ groups the customers with identical choice behavior aiming to buy products $C_l \subseteq J$, also called the consideration set and containing $n_l = |C_l|$ products. These customers arrive over the reservation period according to a Poisson process with a uniform ratio λ_l. The choice behavior is defined by the probability $P_l(j|S)$ that segment l buys product j among the offer $S \subseteq J$. We focus on preference-list choice behavior, which is nonparametric. It is characterized by distinct ordered products with index $l_j \in [1, n_l]$ or 0 if $j \notin J_l$. Conversely, the product $l^k \in J_l$ is the k^{th} product of the preference list if $k \in [1, n_l]$. The subset $C_l^k \subseteq J_l$ with $k \in [1, n_l]$ corresponds to the preference list limited to the first k products. A transition θ_l^k with $k \in [1, n_l]$ reflects the ratio of customers passing from one product to the next in the preference list. Customers always choose a product according to

the order defined by the preference list. We therefore have:

$$P_l(j|S) = \begin{cases} \prod_{k=1}^{l_j} \theta_l^k & \text{if } S \cap C_l^{l_j} = \{j\} \\ 0 & \text{otherwise.} \end{cases}, \quad \forall l \in L,\ j \in J,\ S \subseteq J.$$

We often shorten the preference-list notation to $l^1 \xrightarrow{\theta_l^1} l^2 \xrightarrow{\theta_l^2} \dots \xrightarrow{\theta_l^{n_l}} l^{n_l}$. The total arrival ratio $\lambda_j(S)$ for a product when an offer is proposed is calculated as follows:

$$\lambda_j(S) = \sum_{l \in L} \lambda_l P_l(j|S), \quad \forall j \in J,\ S \subseteq J.$$

We denote by $\lambda(S) = \{\lambda_j(S)\}_{j \in J}$ the arrival ratio vector if offer S is offered.

4.3.1 Dynamic programming formulation

The CNRM problem can be formulated exactly as a DP. We choose a step size h sufficiently small that there is at most one arrival between t and $t + h$. We also introduce x, the vector of remaining capacities ($x = c$ when $t = 0$). The Bellman equations can then be written as follows:

$$\begin{aligned} V(t,x) = V(t+h,x) & \\ + \max_{S \subseteq J(x)} \sum_{j \in S} \lambda_j(S)\, h\, (r_j - \Delta V_j(t+h,x)) & \end{aligned} \tag{DP}$$

where $\Delta V_j(t,x) = V(t,x) - V(t,x - A_j)$ is the opportunity cost of selling product j at time t. $J(x)$ is the set of products with remaining resource capacity. The boundary conditions are:

$$\begin{aligned} V(t,0) = 0, \quad \forall t \in [0,T], \\ V(T+h,x) = 0, \quad 0 \le x \le c. \end{aligned}$$

The optimal policy, denoted by $S^\star(t,x)$, for deciding the availability of each product over the reservation period is formed by the maximization problems solution of S at each time and for each remaining capacity in the previous Equation DP.

Unfortunately, this DP rapidly becomes intractable as the size of the network increases. Even an instance with only ten resources of capacity 100 has 100^{10} states. The CNRM problem must therefore be solved approximately.

4.3.2 Static approximations

We first consider static approximations. They avoid the discrete customer-arrival complexity of the DP by considering a continuous and deterministic flow of customers. All these approximations have the same structure:

$$\begin{aligned} R = \max_{q} \quad & r^\top q \\ s.t. \quad & Aq \leq c, \quad (\pi) \\ & q \geq 0. \end{aligned} \tag{STATIC}$$

where $q = \{q_j\}_{j \in J}$ is the vector of product bookings under a certain demand and policy. The objective function maximizes the revenue, and the constraints ensure that the capacities are respected. An immediate policy is the product booking (PB) that sets the sales limit to the optimal $q_j^\star$ for each product as follows:

$$S^{\text{PB}}(t, x) = \{j \mid q_j \leq q_j^\star\}, \quad \forall t \in [0, T],\ x \leq c. \tag{PB policy}$$

This policy is therefore static because it is fixed for the entire reservation period.

The most popular static approximation is the CDLP (Liu et van Ryzin, 2008), which is based on:

$$\begin{aligned} q = \quad & \sum_{S \subseteq J} \lambda(S) d_S \\ s.t. \quad & \sum_{S \subseteq J} d_S \leq T, \\ & d_S \geq 0, \quad \forall S \subseteq J. \end{aligned} \tag{CDLP}$$

where d_S indicates for how much time each offer should be available. Practitioners derive the offer period (OP) policy by ordering these durations over the reservation period, such that:

$$S^{\text{OP}}(t, x) = \{j \mid j \in S, t \in [t_S, t_S + d_S[\}, \quad \forall t \in [0, T],\ x \leq c. \tag{OP policy}$$

Where t_S depends on how offers are ordered. The different orders are equivalent in theory. As the PB policy, it does not change over the reservation period and is thus static.

To be noted that we can solve the static approximation several times over the reservation period to obtain a more “dynamic" PB or OP policy.

4.3.3 Dynamic approximations

The second type of approximations estimates the pseudo-revenue $r_j - \Delta V_j(t+h, x)$ of each product without solving the entire DP. Most of these approaches implement a decomposition by resource to reduce the number of states. For example, Bront *et al.* (2009) approximate the network value function for resource i as:

$$V(t, x) \approx V_i(t, x_i) + \sum_{k \neq i} \pi_k^\star x_k \tag{DCOMP}$$

where the dual prices $\pi_k^\star$ come from the optimal solution of a static approximation. By substituting this expression into the DP we obtain one DP per resource for the calculation of $V_i(t, x)$. The network opportunity cost $\Delta V_j(t, x)$ can then be approximated by $\Delta \widetilde{V}_j(t, x)$ based on the decompositions by resource, for example (Bront *et al.*, 2009):

$$\Delta V_j(t, x) \approx \Delta \widetilde{V}_j(t, x) = \sum_{\substack{i \in I \\ a_{ij}=1}} \beta \Delta V_i(t, x_i) + (1 - \beta)\pi_i^\star.$$

Where $\Delta V_i(t, x) = V_i(t, x_i) - V_i(t, x_i - 1)$ and $0 \leq \beta \leq 1$ is a parameter to fine-tune. Other approximations have been proposed by Zhang et Weatherford (2017) and Erdelyi et Topaloglu (2010). Similarly to the DP, the policy for the products availability over the reservation period is called the offer dynamic (OD) and is obtained as follows:

$$S^{\mathrm{OD}}(t, x) = \arg \max_{S \subseteq J(x)} \sum_{j \in S} \lambda_j(S) h \left(r_j - \Delta \widetilde{V}_j(t + h, x) \right), \quad \forall t \in [0, T],\ x \leq c. \tag{OD policy}$$

This approach is dynamic because it changes depending on the arrivals.

4.4 Closing approximation

We propose a new static approximation for the CNRM problem under non-parametric choice behavior. Our approximation is based on a new policy, which we call product closing (PC), that is suitable for use with a preference list. It determines the time $t_j \in [0, T]$ when each product becomes unavailable such that the policy is:

$$S^{\mathrm{PC}}(t, x) = \{j \in J \mid t \leq t_j\}, \quad \forall t \in [0, T],\ x \leq c. \tag{PC policy}$$

In other words, it closes the sale of the product at this time.

4.4.1 Buying logic under closing policy

To determine the product sales generated by a PC policy, we start by calculating for how long each segment buys each of its choices. We first note that the k^{th} choice in a preference list is bought provided the product l^k is available, and the products l^h of the previous choices $h \in [1, k[$ are not available. To explain the buying logic driven by the PC policy, we consider $X \xrightarrow{.9} Y \xrightarrow{.8} Z$ as a segment example where X, Y, and Z are three distinct products. In Figure 4.1 we illustrate two cases (a) and (b) of buying logic depending on the PC times for the segment. In case (a), the order is $t_X \leq t_Y \leq t_Z$, i.e., the segment buys X during t_X, then

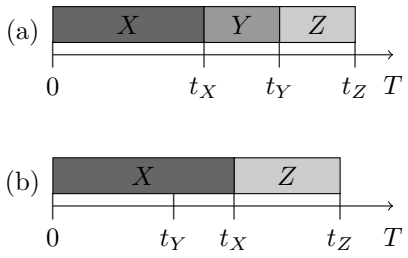

Figure 4.1 Buying logic examples for a segment with preference list $X \xrightarrow{.9} Y \xrightarrow{.8} Z$.

Y during $t_Y - t_X$, and finally Z during $t_Z - t_Y$. In case (b), the order is $t_Y \leq t_X \leq t_Z$, i.e., the segment buys X during t_X and then Z during $t_Z - t_X$ because choice Y is covered by choice X as a consequence of $t_Y \leq t_X$.

To generalize the buying logic, we note that the k^{th} choice is bought if and only if its PC t_{l^k} is greater than the PCs t_{l^h} of the previous choices $h \in [1, k[$. If this condition is satisfied, the choice is bought during the maximum closing $\max\limits_{h \in [1,k[} t_{l^h}$ of the previous choices and its PC t_{l^k}. We can therefore determine the sales duration d_l^k for each segment l and choice k as follows:

$$d_l^k = \left(t_{l^k} - \max_{h \in [1,k[} t_{l^h} \right)^+, \quad \forall k \in [1, n_l],\ l \in L. \tag{4.1}$$

If we apply this formula to the above example, we find the same durations.

4.4.2 Product Closing Program (PCP)

We first simplify the previous formula. For a set S of products, let $t_S = \max\limits_{j \in S} t_j$. These maximum PCs also contain each product PC ($t_j = t_{\{j\}}$). With this notation we can reformulate (4.1) equivalently as:

$$d_l^k = t_{C_l^k} - t_{C_l^{k-1}}, \quad \forall k \in [1, n_l],\ l \in L.$$

The quantity that the segment buys is then obtained by multiplying this duration by the buying probability as defined in Section 4.3. We can then write the PC program (PCP) as the following approximation:

$$
\begin{aligned}
q_j = \quad & \sum_{l \in L} \lambda_l \prod_{k=1}^{l_j} \theta_l^k d_l^{l_j}, && \forall j \in J, && \text{(PCP)} \\
s.t. \quad & d_l^k = t_{C_l^k} - t_{C_l^{k-1}}, && \forall k \in [0, n_l],\ l \in L, \\
& t_S = \max_{j \in S}\ t_j, && \forall S \in C_L, \\
& t_j \in [0, T], && \forall j \in J.
\end{aligned}
$$

where C_L is the union of the segment consideration subsets, determined as follows:

$$C_L = \bigcup_{l \in L} \bigcup_{k=1}^{n_l} C_l^k.$$

For example, for two segments with preference lists $X \to Y \to Z$ and $Y \to X$, respectively, C_L is $\{\{X, Y\}, \{X, Y, Z\}\}$. The number of sets n_L corresponding to the cardinality of $|J_L|$ depends on the number of segments, the number of products considered, and the overlap between segments. A simple analysis allows us to bound n_L between $\max_{l \in L}\ n_l - 1$ when the segments overlap completely and $\sum_{l \in L} n_l - 1$ when there is no overlap.

4.4.3 Quality of the PCP

In this section, we compare our PCP approximation to the exact DP formulation and the CDLP approximation. We start by comparing the PC policy to the OP policy derived from the CDLP:

Proposition 1. *A PC policy always has a unique equivalent OP policy denoted by OP_{PC}.*

Proof. Each PCs $\{0 \leq t_j \leq T\}_{j \in J}$ can be ordered by time. We thus obtain n periods, indexed by $k \in [1, n]$. These periods start at t_{k-1} and finish at t_k, with $t_0 = 0$, and their durations are $d_k = t_k - t_{k-1}$, with $d_0 = 0$. The OP_{PC} policy is thus defined by $S_k = \{j \mid t_j \geq t_k\}$ A period with a null duration reflects products sharing a same closing time. The uniqueness of the equivalent OP is immediate. □

With the previous proposition, we next prove that the optimal revenue returned by the CDLP is always an upper bound on the optimal PCP revenue.:

Proposition 2. *$R_{PCP}(PC) = R_{CDLP}(OP_{PC})$ and thus $R^\star_{\text{PCP}} \leq R^\star_{\text{CDLP}}$.*

Proof. We use the definitions of Proposition 1 proof for d_k, t_k and S_k of the OP_{PC} policy. Note that $\sum_{k=0}^{n} d_k = t_n \leq T$ satisfies the second constraint of the CDLP. We calculate the product quantity sold in CDLP via:

$$q_j^{\text{CDLP}} \quad = \sum_{k=1}^{n} \lambda_j(S_k) d_k = \sum_{l \in L} \lambda_l \sum_{k=0}^{n} P_l(j|S_k) d_k.$$

For any product and segment, it exists $s_{j,l}$ and $e_{j,l}$ in $[0, n]$ corresponding to the period indexes when the segment respectively starts and ends buying the product such that:

$$q_j^{\text{CDLP}} = \sum_{l \in L} \lambda_l \sum_{k=s_{j,l}}^{e_{j,l}} P_l(j|S_k) d_k = \sum_{l \in L} \lambda_l \prod_{h=1}^{l_j} \theta_l^h \left(t_{e_{j,l}} - t_{e_{j,l}-1} + t_{e_{j,l}-1} - \cdots - t_{s_{j,l}} \right)$$

By simplification of the sum and because $t_{s_{j,l}} = t_{C_l^{l_j-1}}$ and $t_{e_{j,l}} = t_{C_l^{l_j}}$, we obtain:

$$q_j^{\text{CDLP}} = \sum_{l \in L} \lambda_l \prod_{h=1}^{l_j} \theta_l^h \left(t_{C_l^{l_j}} - t_{C_l^{l_j-1}} \right) = \sum_{l \in L} \lambda_l \prod_{h=1}^{l_j} \theta_l^h d_l^{l_j} = q_j^{PCP}.$$

Therefore, OP_{PC} is a feasible solution for CDLP and $R_{\text{PCP}}(PC) = R_{\text{CDLP}}(OP_{\text{PC}})$. □

In CDLP, $R^\star_{\text{DP}} \leq R^\star_{\text{CDLP}}$, but this is not the case for PCP because it ensures nonreopening. We sell each product until a specified time and then never sell it again:

Lemma 3. *Nonreopening* $\Leftrightarrow \forall j \in S_t \mid \forall t' < t$ *then* $j \in S_{t'}$ *with* $t \in [0, T]$.

A no-reopening policy is sometimes mandatory in practice. The PC policy prohibits reopening, whereas OP and OD do not if no constraints are added. Thus, the optimal PCP revenue could be less than that for DP.

We now prove that PCP and CDLP are equivalent when there nonreopening.

Proposition 4. *If nonreopening, A OP policy has a unique equivalent PC policy denoted* PC_{OP}.

Proof. By definition, every product has a unique closing time if nonreopening. PC_{OP} is thus defined by $t_j = \sum_{k \in [1,n] \mid j \in S_k} d_k$. □

The nonreopening case allows us to conclude several properties for the PCP:

Proposition 5. *If nonreopening, PCP and CDLP are equivalent. By inheritance* $R^\star_{\text{DP}} \leq R^\star_{\text{PCP}}$ *and PCP is asymptotically optimal.*

Proof. With the equivalent OP_{PC} we prove similarly to Proposition 1 that $R_{PCP}(OP) = R_{CDLP}(PC_{\mathrm{OP}})$ and thus $R^{\star}_{PCP}(OP) \geq R^{*}_{CDLP}(PC_{\mathrm{OP}})$. With Proposition 2, we obtain $R^{\star}_{PCP} = R^{*}_{CDLP}$. The inherited properties come from the results on CDLP proved by Liu et van Ryzin (2008). □

4.5 Solving the PCP

In this section, we describe how we linearize the PCP to obtain a mixed integer linear program. We also present methods to rapidly solve the linearization.

4.5.1 Linearization

Our approximation is nonlinear because of the constraint $t_S = \max_{j \in S} t_j$, which appears n_L times. It can be linearized by adding binary variables. We introduce the following binary variables, also called hierarchy variables:

$$h_g^f = \begin{cases} 1 & \text{if } t_f > t_g, \quad \forall f, g \in J \\ 0 & \text{otherwise.} \end{cases}$$

Each hierarchy variable equals one if y is open for longer than z and zero otherwise. We naturally have $h_g^f = 1 - h_f^g$ and we assume $h_f^f = 1$ for all $f, g \in J$.

We note that for any set $S \in C_L$, there always exists at least one $l \in L$ and $k \in [1, n_l]$ such that $S = C_l^k$ which is a segment sub consideration set. We define $\widehat{S}$ a direct subset of S with cardinality $|S| - 1$ as follows:

$$\exists l \in L, k \in [1, n_l] \mid \widehat{S} = C_l^{k-1},$$

We denote $\widetilde{j} = l^k$ such that $S = \widehat{S} \cup \widetilde{j}$. There always exits at least one other subset $\widetilde{S} \subset S$ define as follows:

$$\exists l \in L, k \in [1, n_l] \mid \widetilde{S} = C_l^k \ \textit{with} \ \ S = \widetilde{S} \cup \widehat{S}$$

The proof is that $\{\widehat{j}\}$ is an admissible subset. Consequently $\widetilde{j} \in \widetilde{S}$ and we can linearize efficiently the closing of S.

Suppose for example that we have the four products $S = C_l^k = \{A, B, C, D\}$. One of the direct subset $\widehat{S} = C_l^{k-1}$ is in this case $\{A, B, C\}$ and thus $\widetilde{j} = D$. Imagine that $\widetilde{S} = \{A, D\}$ is another admissible subset. We have $t_S = t_{\widetilde{S}}$ if and only if D is open for longer than B and C. Conversely, $t_S = t_{\widehat{S}}$ if and only if B or C are open for longer than D. The fact that product A is shared by both subsets reduces the number of hierarchy verifications.

This leads to the PC mixed integer program (PCMP) with the previous definition of $\widehat{S}$ and $\widetilde{S}$:

$$
\begin{aligned}
q_j = & \sum_{l \in L} \lambda_l \prod_{k=1}^{l_j} w_l^k d_l^{l_j} && \forall j \in J && \text{(PCMP)}\\
s.t. \quad & d_l^k = t_{S_l^k} - t_{S_l^{k-1}} && \forall k \in [0, n_l],\ l \in L\\
& t_S \geq t_{\widehat{S}} && \forall S \in C_L\\
& t_S \geq t_{\widetilde{S}} && \forall S \in C_L\\
& t_S \leq t_{\widehat{S}} + T \sum_{j \in \widehat{S}, j \notin \widetilde{S}} h_j^{\widetilde{j}} && \forall C \in C_L\\
& t_S \leq t_{\widetilde{S}} + T h_{\widetilde{j}}^j && \forall j \in \widehat{S},\ j \notin \widetilde{S},\ S \in C_L\\
& h_g^f \in \{0,1\} && \forall f, g \in J
\end{aligned}
$$

To limit the number of constraints, we must find the set $\widetilde{S}$ with the highest cardinality. In fact there is $2 \times (1 + |S| - |\widetilde{S}|)$ constraints per linearization. We do this when building the program, and we exploit the overlap between segments. Our model uses overlap to eventually reduce complexity.

4.5.2 Use of hierarchy

The hierarchy variables represent a hierarchy between products that could be fixed before we solve the PCMP. This leads to the PC linear program (PCLP) for any fixed hierarchy $\widehat{h}$:

$$
\begin{aligned}
q_j = & \sum_{l \in L} \lambda_l \prod_{k=1}^{l_j} \theta_l^k d_l^{l_j} && \forall j \in J, && \text{(PCLP)}\\
s.t. \quad & d_l^k = t_{C_l^k} - t_{C_l^{k-1}} && \forall k \in [0, n_l],\ l \in L\\
& t_S = t_j \mid \widehat{h}_f^j = 1, \forall f \in S && \forall S \in C_L
\end{aligned}
$$

For the optimal hierarchy, the PCLP and PCMP are equivalent. However, there are $n!$ permutations of the products, and each one is an admissible hierarchy. Determining the optimal hierarchy is thus a difficult combinatorial problem.

It is easier to find a good but not necessarily optimal hierarchy. We can for example:

— Rank products by price;

— Rank products by price divided by number of resources;

— Reuse a hierarchy from a previous PCMP optimal solution;

— Use a hierarchy specified by the company (often called nesting in practice).

Solving the PCLP with a good but not optimal hierarchy gives a solution that can be useful. We can use it to speed up the solution of the PCMP branch-and-bound algorithm. The solution may also be useful if the PCMP is too large to be solved in the time available.

4.5.3 Reopening (CDPC)

Our PCP approximation does not allow reopening but the corresponding PC solution can always be transformed to an equivalent OP policy (OP_{PC}) according to Proposition 1 of Section 4.4.3. So that it provides a good initial solution for any CDLP column generation algorithm because $R_{\text{CDLP}}(OP_{\text{PC}}) = R_{\text{PCP}}(PC)$ according to Proposition 2 of Section 4.4.3. It also allows us to "reopen" our PCP solution. We call this approach the Choice Deterministic with Products Closing initial solution (CDPC).

4.6 Numerical experiments

In this section, we conduct numerical experiments to benchmark the following approximations:

CDLP (Liu et van Ryzin, 2008): Described in Section 4.3.2 and solved by column generation with the Hosseinalifam *et al.* (2016) subproblem for preference-list choice behavior.

SDCP (Meissner *et al.*, 2013): We add product constraints for larger subsets until the objective function no longer changes.

PCLP: Presented in Section 4.5.2. The hierarchy is established by ranking products by their price and then by their potential demand if price are equals. The hierarchy is obtained by ranking the products by price ($\widehat{h}_g^f := r_f > r_g$).

PCMP: Presented in Section 4.5.1. An initial solution corresponding to the previous PCLP solution is given to the branch-and-bound process, as explained in Section 4.5.2. The relative integrability gap is set to 10^{-3}.

CDPC: The CDLP approximation with an initial solution given by the PCMP, as explained in Section 4.5.3.

We use the following policies:

OP is the OP policy described in Section 4.3.2. It is obtained by a lexicographic sequencing of the CDLP durations d_S.

PB is the PB policy corresponding to fixing a static limit q_j for each product, as explained in Section 4.3.2.

PC is the PC policy returned by the PCP, as explained in Section 4.4.

OD is the OD policy described in Section 4.3.3. It is obtained by the dynamic decomposition of Bront *et al.* (2009) with $\beta = 1$.

The quality of an approximation depends on its solution time denoted by CPU and the expected revenue E[R]. We use a Monte-Carlo approach with a discrete-arrival simulation to determine the expected revenue. We generate random discrete arrivals by generating arrival timings according to a Poisson process for each segment. Each simulation is stopped after a number of evaluations specific to the instance.

We build scenarios by varying the load factor LF. The load factor is simply the sum of arrivals over the sum of capacities: $\text{LF} = \sum_{l \in L} \lambda_l / \sum_{i \in I} c_i$. By multiplying all the λ_l by the same factor, we obtain the desired LF.

We define the capacity factor as the percentage of remaining capacity: $\text{CF} = \sum_{i \in I} x_i / \sum_{i \in I} c_i$.

4.6.1 Parallel flights

Our first instance, parallel flights, is illustrated in Figure 4.2. It is composed of three parallel flights, of capacity 100, from city A to city B at 09:00, 11:00, and 20:00. We consider two

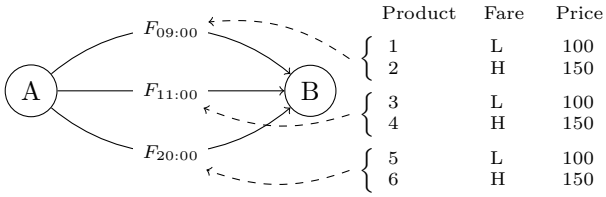

Figure 4.2 Resources and products for parallel flights.

fares H (150) and L (100) per flight, giving six products. The reservation period lasts 360 periods. The customers are divided into four segments, as shown in Table 4.1.

Table 4.1 Segments for parallel flights.

Segment	Arrival ratio	Choice behavior
1	0.17LF	$1 \xrightarrow{0.89} 2$
2	0.25LF	$1 \to 3 \to 5 \xrightarrow{0.89} 2 \to 4 \to 6$
3	0.17LF	$3 \to 5 \xrightarrow{0.89} 4 \to 6$
4	0.25LF	$5 \xrightarrow{0.89} 6 \xrightarrow{0.87} 3 \xrightarrow{0.89} 4$

Table 4.2 presents the running time and expected revenue for the parallel flights instance. We first note that approximations return very similar results for a same policy. It means that

the three approximations are really similar as we can see in Table A.2 of the e-companion where approximations share the same ideal revenue and capacity factor.

If we now focus on policies, we observe that OD almost always performs better than others in terms of expected revenue. In average, it is 1.4% better than the CDLP-OP reference whereas PB and PC are respectively -0.3% and 0.1%. In fact, it is the only dynamic policy and it takes into account the order of arrivals contrarily to the three other static policies OP, PB and PC. We can also see the effect of the dynamic aspect in Figure 4.3 where the OD policy often has the highest expected capacity factor meaning that it captures more bookings. It also explains why the OD policy performs better in comparison with other policies when the load factor is low or high.

Table 4.2 Running seconds CPU and expected revenue E[R] for Parallel flights by simulation.

		CDLP			SDCP		PCMP		
LF		OP	PB	OD	PB	OD	PC	PB	OD
0.6	CPU	0.23	0.23	4.50	0.05	4.21	0.05	0.03	4.17
	E[R]	23918±0.28%	22995±0.18%	23925±0.28%	22997±0.17%	23915±0.28%	23887±0.28%	23003±0.17%	23904±0.28%
	ΔE[R]	–	-3.86	0.03	-3.85	-0.01	-0.13	-3.83	-0.06
0.8	CPU	0.03	0.03	6.30	0.03	6.55	0.03	0.03	6.71
	E[R]	31312±0.3%	30725±0.2%	31399±0.3%	30717±0.2%	31377±0.3%	31221±0.3%	30737±0.2%	31414±0.3%
	ΔE[R]	–	-1.36	0.55	-1.55	0.60	-2.52	-1.66	0.70
1.0	CPU	0.55	0.55	9.10	0.03	8.60	0.02	0.02	8.87
	E[R]	37152±0.2%	37083±0.1%	37650±0.2%	37119±0.2%	37596±0.2%	37780±0.3%	37186±0.2%	37669±0.2%
	ΔE[R]	–	0.04	2.29	0.17	2.14	1.92	0.28	2.22
1.2	CPU	0.27	0.27	10.95	0.02	10.78	0.02	0.02	10.80
	E[R]	43318±0.2%	43436±0.1%	43575±0.2%	43359±0.1%	43452±0.2%	43336±0.2%	43367±0.1%	43455±0.2%
	ΔE[R]	–	0.36	0.81	0.44	1.05	0.41	0.36	0.71
1.4	CPU	0.03	0.03	12.85	0.02	13.11	0.04	0.04	13.13
	E[R]	43481±0.11%	44990±0.01%	44930±0.02%	44987±0.01%	44942±0.01%	43837±0.11%	44991±0.01%	44991±0.01%
	ΔE[R]	–	3.47	3.33	3.46	3.36	0.82	3.47	3.47
$\overline{\text{CPU}}$		0.22	0.22	8.74	0.03	8.65	0.03	0.03	8.74
$\overline{\Delta\text{E[R]}}$		–	-0.27	1.40	-0.27	1.43	0.10	-0.28	1.41

ΔE[R] is the relative difference with respects to OP policy given by the CDLP-OP.
Simulation has 3000 evaluations.

However, when we compare the running time, the OD policy is by far the slowest whereas OP, PC and PB are equivalent. The latest policy are in average 30 times faster than the OD policy for this instance. This long running time comes from its building process as we can see in the Table A.2 of the e-companion where the time for building each policy is reported. This is mainly due to the high number of dynamic program to solve as we explained in Section 4.3.3. Moreover, this building time increases with the load factor because it depends on the number of arrivals, the capacities and the number of resources.

This example also highlights the really unequal performance of the PB policy with respect to the load factor. It is outperformed by the CDLP-OP for LF inferior or equal to one but up to

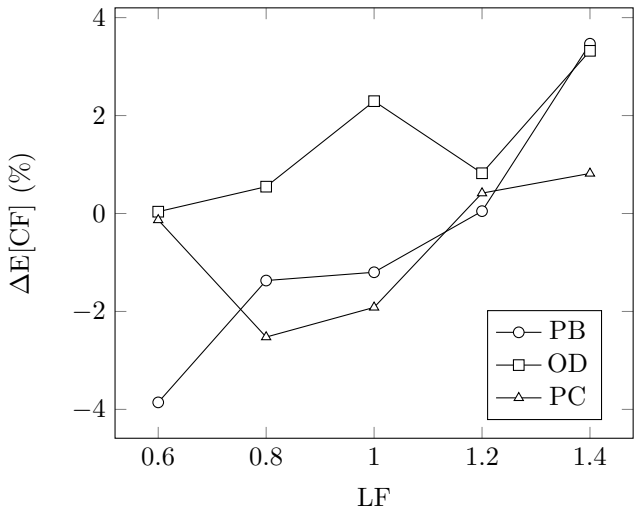

Figure 4.3 Expected capacity factor relative difference ΔE[CF] with respects to CDLP-OP for Parallel flights.

3% better for higher load factor. This is due to the fact that PB policy capture exactly the number of bookings provided by the related approximation. Such that when the load factor is inferior to one, it will never capture any eventual additional demand even if capacities are not reached. It also explains that the capacity factor is really low when the load factor is inferior to one in Figure 4.3 and in comparison of the other policies. Nonetheless, when the load factor is up to one, the PB policy becomes a really efficient policy because capacities are reached in the approximation.

One important fact regarding the SDCP approximation is that it cannot return an OP policy even if it is built on offers duration. In fact, the products constraints added, as explained in Meissner *et al.* (2013), do not ensure homogenized durations across segments. For $LF = 1$, the second segment offers $\{3, 5\}$ and $\{1, 3, 5\}$ respectively during 89.4 and 270.4 periods while third segments offers $\{3, 5\}$ during 360 periods. Products constraints are respected but we cannot conclude offers duration shared by every segment. That is why the SDCP solution is only used to build PB and OD policy for numerical experiments.

4.6.2 Bus-line instance

The bus-line instance has two buses leaving at 07:00 and 11:00 from city A to cities B, C, and D. Six markets are thus served, as illustrated in Figure 4.4. Each bus has a capacity of 30 and there are $2 \times 3 = 6$ resources. Two fares (low, high) are offered for each trip, giving a total of $6 \times 2 \times 2 = 24$ products. In the bus industry, tickets are usually available at least two months in advance, so we set $T = 60$ days. We consider five segments each considering

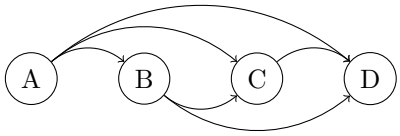

Figure 4.4 Markets for Bus-line instance.

4 products. In total there are $3 \times 6 = 18$ segments. A complete description of the instance is given in the e-companion at A.1.

Table 4.3 shows the running time and expected revenue for the Bus-line instance. We come to the same conclusions as for the previous Parallel flights instance concerning the equivalence of approximations. We note that the performance of the PB, PC and OD policies over the CDLP-OP improves as load factor increases. For the PB policy, the reason is the same as for the Parallel flights instance and is explained in Section 4.6.1. PC is a more robust policy than OP when there is nonreopening. The dynamic aspect of OD ensures better expected revenue than other policies. These respective qualities of PC and OD are emphasized when the load factor increases because the policy is more selective contrarily to a low load factor for which most of the demand is accepted.

Table 4.3 Running seconds CPU and expected revenue E[R] for Bus-line by simulation. ΔE[R] is the relative difference with respects to OP policy given by the CDLP-OP. Simulation has 1000 evaluations.

		CDLP			SDCP		PCMP		
LF		OP	PB	OD	PB	OD	PC	PB	OD
	CPU	0.89	0.89	639.01	0.08	701.42	0.05	0.05	565.79
0.6	E[R]	12413±0.40%	12039±0.43%	12452±0.41%	12025±0.43%	12460±0.38%	12468±0.44%	12004±0.41%	12443±0.39%
	ΔE[R]	–	-3.01	0.31	-3.12	0.38	0.44	-3.29	0.24
	CPU	1.16	1.16	936.45	0.09	994.02	0.06	0.06	857.59
0.8	E[R]	14909±0.39%	14668±0.35%	15052±0.33%	14678±0.34%	15075±0.33%	14958±0.39%	14655±0.34%	15038±0.34%
	ΔE[R]	–	-1.62	0.96	-1.55	1.11	0.33	-1.70	0.87
	CPU	0.74	0.74	794.38	0.08	811.36	0.05	0.05	795.55
1.0	E[R]	16496.3±0.36%	16528±0.32%	16971±0.31%	16543±0.32%	16965±0.31%	16736±0.37%	16492±0.33%	16979±0.31%
	ΔE[R]	–	0.19	2.88	0.28	2.84	1.45	-0.03	2.92
	CPU	0.89	0.89	1008.38	0.09	882.67	0.05	0.05	866.67
1.2	E[R]	17516±0.35%	17883±0.31%	18120±0.24%	17879±0.30%	18186±0.23%	17747±0.37%	17854±0.29%	18097±0.24%
	ΔE[R]	–	2.09	3.45	2.08	3.83	1.32	1.93	3.32
	CPU	0.95	0.95	1319.33	0.10	1038.75	0.03	0.03	992.62
1.4	E[R]	18179.3±0.35%	18848±0.28%	19114±0.25%	18858±0.29%	19133±0.25%	18582±0.35%	18839±0.28%	19104±0.26%
	ΔE[R]	–	3.68	5.14	3.73	5.24	2.21	3.63	5.08
$\overline{CPU}$		0.92	0.92	939.51	0.09	885.64	0.05	0.05	5.52
$\overline{\Delta E[R]}$		–	0.27	2.55	0.28	2.68	1.15	0.11	2.49

This example underlines the good performance of the SDCP which is solved in average 5 times faster than the CDLP. The products closing constraints added are sufficient to return the same optimal revenue as reported in Table A.3 of the e-companion. We cannot build OP

policy but the policies PB and OD derived perform as well as or better than the CDLP ones for any load factor.

It is clear in Table 4.3 that building the OD policy requires important postprocessing, as explained in Section 4.3.3, and thus considerable time. Table A.3 confirms that almost all the running time is spent on building the policy and not in solving the approximation. Even if a leg decomposition is used, a mathematical program must be solved per leg $i \in I$ for each remaining capacity c_i and each potential arrival $\sum_{i \in I} c_i \times \text{LF}$. Therefore, the number N_{OD} of values to find and store for the OD policy is:

$$N_{\text{OD}} = \sum_{i \in I} c_i \left(\sum_{l \in L} \lambda_l T \right)$$

The bus-line instance is relatively small, but N_{OD} is already equal to $6 \times 30 \times (6 \times 30LF) = 32400LF$. It explains why the running time increases when the load factor augments as observed at Table 4.3.

To investigate the OD tractability, we complicate the initial instance progressively and report the number of values N_{OD} and the time needed to build this policy at Table 4.4. The OD

Table 4.4 Time CPUp to build the OD policy for Bus-line cumulative changes.

Cumulative changes	N_{OD}/LF	CPUp
Initial instance	3.2×10^4	11min
+ Two more buses per day of capacity 30	1.3×10^5	1h43
+ Line has two more cities E and F	3.6×10^5	18h50
+ Capacity of buses pass from 30 to 300 (train)	3.6×10^7	53h05

policy is without doubts the best but become rapidly intractable when instances grow. Each value to find is often obtained by solving a complex model as explained in Section 4.3.3. And also because computationally it is a lot of values to store. In practice, the reservation systems may not support this amount of data for a complete network.

4.6.3 Airline instance

The airline instance is based on the Delta Air Lines network limited to eight major US airports, as illustrated in Figure 4.5. We start by limiting the instance on the five largest airports: ATL, LAX, ORD, DFW, and DEN. A complete description of the instance is given in the e-companion at A.1.

We do not benchmark the OD policy for this instance because the problem become intractable for this size, as shown in Section 4.3.1 and confirmed by tests. For the SDCP, the number

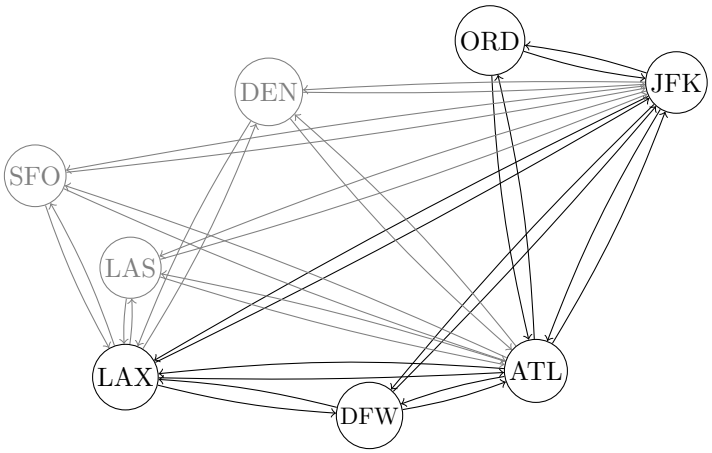

Figure 4.5 Markets of Airline. The five largest airports are represented in bold.

of products constraints is at most $\binom{L}{2} \times 2^{\max_{l \in L} n_l} = 95703 \times 1024 \approx 9.8 \times 10^7$ according to Meissner *et al.* (2013). Even if this is an upper bound, the search for the intersections between segments is intractable. That is why we do not benchmark the SDCP in the Airline instance. The CDLP with column generation takes much time to solve and PCMP resolution is more difficult. We thus introduce the CDPC and PCLP approximations for this larger instance.

Table 4.5 Running seconds CPU and expected revenue E[R] for Airline by simulation.ΔE[R] is the relative difference with respects to OP policy given by the CDLP-OP. Simulation has 500 evaluations.

		CDLP		PCMP		CDPC		PCLP	
LF		OP	PB	PC	PB	OP	PB	PC	PB
0.6	CPU	5054.17	5054.74	5.84	5.86	1652.13	1652.87	3.86	3.84
	E[R]	1286871±0.07%	1255551±0.06%	1290783±0.08%	1255436±0.06%	1286643±0.07%	1255663±0.06%	1277509±0.07%	1244115±0.06%
	ΔE[R]	–	-2.43	0.30	-2.44	-0.02	-2.43	-0.73	-3.32
0.8	CPU	6212.84	6213.59	8.26	8.26	2601.45	2601.26	4.43	4.43
	E[R]	1531298±0.06%	1506555±0.05%	1536923±0.06%	1509048±0.05%	1531944±0.06%	1506016±0.05%	1522267±0.06%	1496016±0.05%
	ΔE[R]	–	-1.62	0.37	-1.45	0.04	-1.65	-0.59	-2.30
1.0	CPU	5095.71	5095.71	12.55	12.57	1583.38	1583.19	5.37	5.37
	E[R]	1714632±0.06%	1698282±0.05%	1718955±0.05%	1701311±0.05%	1716096±0.06%	1697962±0.05%	1704529±0.05%	1687989±0.05%
	ΔE[R]	–	-0.95	0.25	-0.78	0.09	-0.97	-0.59	-1.55
1.2	CPU	4648.74	4649.62	7.95	7.95	1692.24	1693.00	4.99	4.99
	E[R]	1862207±0.05%	1851161±0.04%	1868050±0.05%	1858369±0.04%	1863181±0.1%	1855136±0.0%	1853486±0.05%	1844254±0.05%
	ΔE[R]	–	-0.59	0.31	-0.21	0.05	-0.38	-0.47	-0.96
1.4	CPU	6651.45	6652.18	9.05	9.05	1800.14	1800.86	5.52	5.52
	E[R]	1991666±0.06%	1985538±0.04%	1996496±0.05%	1992705±0.04%	1992173±0.06%	1985050±0.05%	1979837±0.05%	1977171±0.04%
	ΔE[R]	–	-0.31	0.24	0.05	0.03	-0.33	-0.59	-0.73
$\overline{\text{CPU}}$		5532.58	5533.17	8.73	8.74	1865.87	1866.10	4.83	4.83
$\overline{\Delta\text{E[R]}}$		–	-1.18	0.29	-0.97	0.04	-1.15	-0.59	-1.72

Table 4.5 reports the running time and expected revenues of the CDLP, PCMP, CDPC and PCLP for the Airline instance with different load factor. The full results are reported in Table A.4 of the e-companion.

We observe the same phenomenon for the PB policy as for the previous instances. It can not capture the excess of the demand which is problematic for low factor and rapidly overshad-

owed by capacity saturation when load factor increases.

We also note that our approach is computed in less than 15 seconds, which is remarkable given the instance size. It is much faster than the CDLP and always returns a slightly better expected revenue. This gain in revenue, in average 0.3%, for the PCMP must be explained by the robustness of closing sales once rather than proposing different offers over the reservation period.

Even though, we note that the CDLP always returns a slightly better optimal revenue in the e-companion at A.4. This may be explained by the integrity gap chosen for PCMP or the reopening permitted by CDLP.

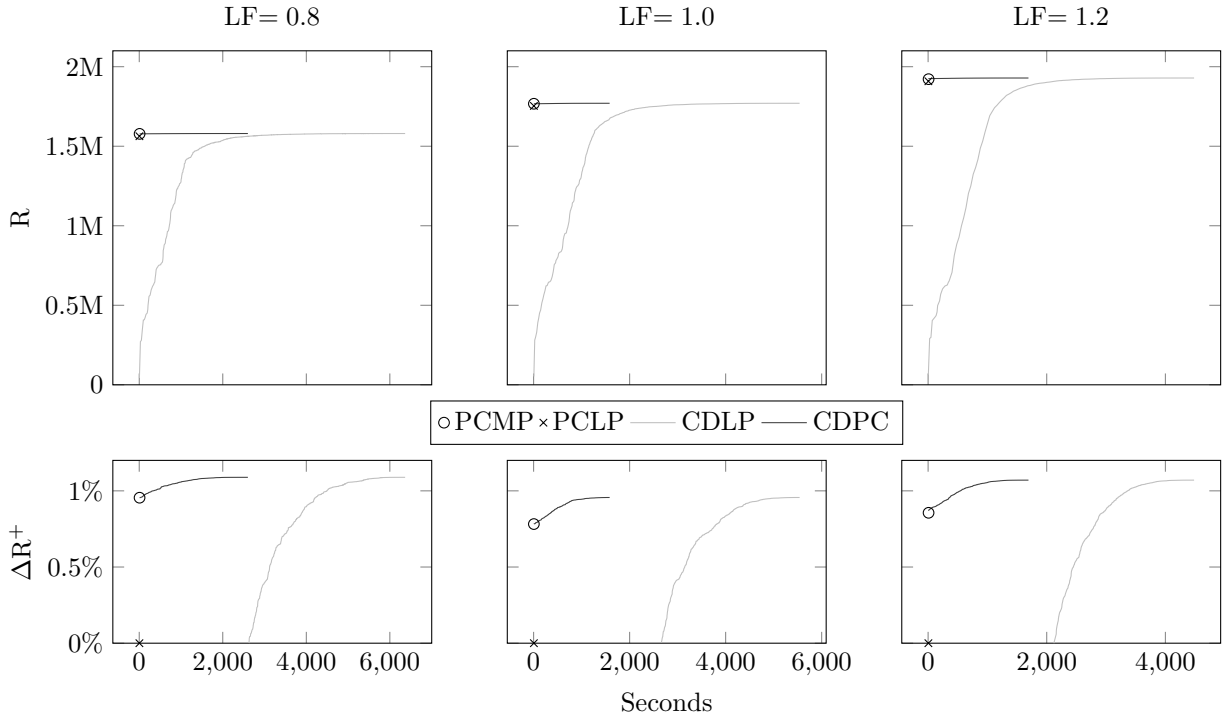

Figure 4.6 Ideal revenue towards time for Airline.

This instance also shows the good quality of our PCLP heuristic. In fact PCLP is solved twice faster than PCMP and returns an expected revenue only 0.59% lower than the CDLP-OP. However, solving PCMP remains quick and the difference in expected revenue with this approximation is almost 1.0%.

We also observe the good performance of our CDPC approach. It accelerates in average by three the CDLP resolution and returns the same ideal revenue (see e-companion Table A.4) and similar expected revenue, as we can see in Table 4.5, with a 0.04% difference. We thus obtain in much less time a really good reopening solution by mixing PCMP and CDLP.

To better illustrate the convergence speed, we plot in Figure 4.6 the optimal revenue R of

each approximation vs. the solution time for different load factor. CDLP and CDPC are plot by cherry piking and smoothing their column generation solving. ΔR^+ is the optimal revenue relative difference in percent with respect to the PCLP when positive.

We observe that our PCMP approximation rapidly returns a near optimal solution contrarily to the CDLP. The latter takes more than one hour to converge to solutions found in average in less than 15 s by PCMP.

The gain in time by choosing the PCMP as a initial solution for the CDLP is perfectly represented in the Figure 4.6. We note that the remaining column generation increases only by less than 0.1% the solution and the convergence is very slow.

To test the tractability of our approach, we now increase progressively the number of cities in the network. Table 4.6 lists the evolution of the network characteristics.

Table 4.6 Airline characteristics by number of cities considered. The five initial cities are ATL, LAX, ORD, DFW, and DEN.

# Airports	Flights	Markets	Products	Segments	Consideration sets
5	115	20	1591	438	$1 \leq 7.93 \leq 10$
6 +DEN	137	30	2724	630	$1 \leq 9.26 \leq 12$
7 +SFO	184	42	4518	896	$1 \leq 10.60 \leq 14$
8 +LAS	220	56	6884	1199	$1 \leq 12.02 \leq 16$

Figure 4.7 reports the running time CPU, on a logarithmic scale, for CDLP, PCMP CDPC and PCLP and the expected revenue E[R] for different sizes of network.

The running times are really similar for the load factors experimented. The faster resolution of the PCMP in comparison with the CDLP is even more pronounced as the network grows. Indeed, the CDLP is far longer to solve because each subproblem highly suffers from the increase of products.

The difference between the CDLP and the CDPC running time is considerable. In fact, it corresponds to the time for the CDLP to reach the PCMP ideal revenue. This shows how much the PCMP convergence (branching the hierarchy binaries) is faster than the CDLP column generation. Moreover, it emphasizes the significant benefice of taking the PCMP as initial solution for the CDLP (CDPC).

Not surprisingly, the expected revenue is higher as the load factor or the number of cities increases. We note that the PCMP returns a slightly better expected revenue (between 0.25% and 0.61%). As for the previous instances, this illustrates the more robust structure of the PC policy.

We observe that, in average, the PCMP is solved in 60 s for 7 cities and in 450 s for 8

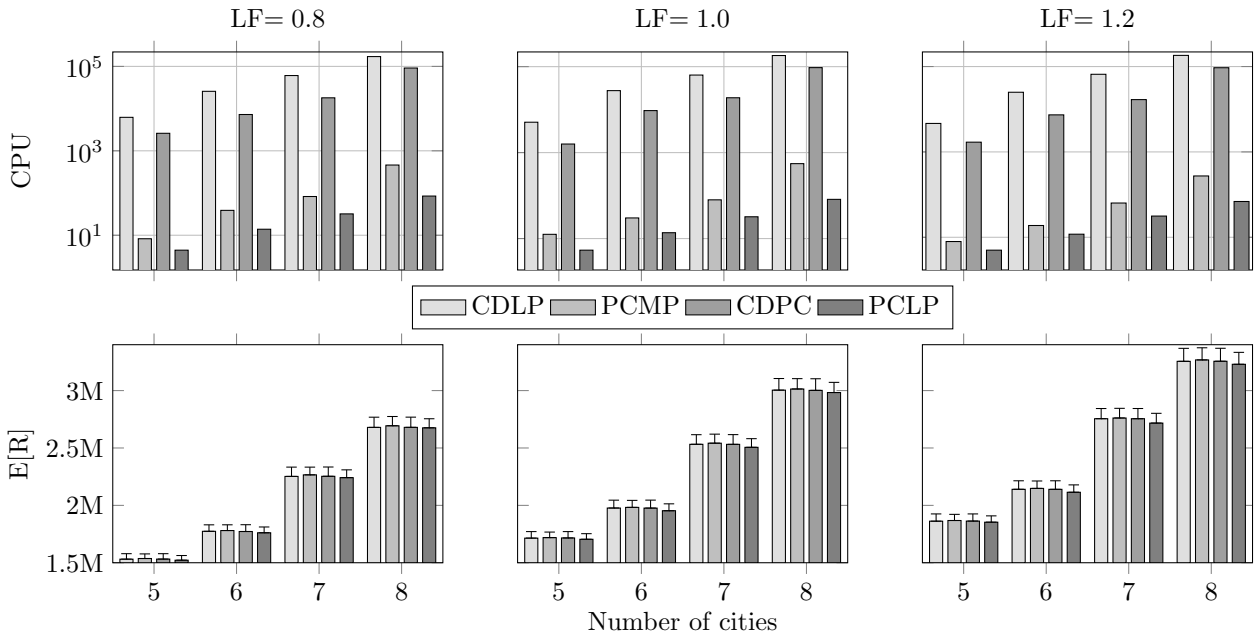

Figure 4.7 Running seconds CPU and expected revenue E[R] for Airline with scaled number of cities considered.

cities. This noticeable gap underlines the first difficulties for the PCMP as instances grows. On another side, the PCLP requires respectively 38 s and 60 s and does not seem not as impacted by this scaling. Its solving time increases smoothly and the expected revenue is only respectively 0.31% and 0.62% lower than CDLP and PCMP. The PCLP seems a good alternative for largest instance and the expected revenue returned could be improved by better method to select the hierarchy.

4.7 Conclusion

We have presented a new static approximation for the CNRM problem with nonparametric choice behavior. We focus on the preference list because the multinomial logit model suffers from the independence of irrelevant alternatives. Rather than working with offers, we work directly with the products and determine when to stop selling each one. For small and medium instances, the different approximations and associated policies (OP, PC, PB, OD) give similar results. However, OD can give the best results if the leg decomposition is appropriate for the instance, because of its dynamic adaptation to the stochastic demand. For larger instances, our approximation outperforms the current approximations because the policy gives a slightly better expected revenue for a much shorter solution time. Our approximation is based on a no-reopening policy. A solution with reopening can be generated

by using the PCMP solution as an initial solution for CDLP. This two-phase approach greatly accelerates CDLP. For even larger instances, our approximation is designed to become linear if a hierarchy is fixed. A good hierarchy is in practice not hard to find. The linear program obtained can be rapidly solved and returns a near-optimal solution. With its greatly reduced solution time and good-quality policy, our approximation is a promising approach for practical implementations.

Acknowlegment

The authors are grateful to the natural sciences and engineering research council of Canada (NSERC), the Fonds de recherche du Quebec en nature et technologies (FRQNT) and the company ExPretio technologies for having funded and supported this research. The authors also thank the anonymous referees and associate editor, whose feedback helped improve the paper.

CHAPITRE 5 ARTICLE 2: BUYING GRAPH FOR CHOICE NETWORK REVENUE MANAGEMENT

Le texte de ce chapitre est celui de l'article *Buying Graph for Choice Network Revenue Management* soumis au journal *INFORMS Operations Research*. Auteurs : Thibault Barbier, Miguel F. Anjos, Fabien Cirinei et Gilles Savard.

Abstract

The current challenge in network revenue management is the incorporation of customer choice behavior. This increases the complexity of the model, but it ensures more robust policies that return better expected revenues when simulated. The resulting choice network revenue management (CNRM) problem has an exact dynamic programming formulation that rapidly becomes intractable. Almost all the CNRM approximations available are designed for choice behavior that is either parametric (multinomial logit) or nonparametric (preference lists). We propose a new CNRM approximation that can accept both parametric and nonparametric models. Our approach is based on a buying graph that represents the possible customer behavior whatever the initial choice behavior. This tree is built and solved iteratively, allowing us to solve even the largest instances within a specified time window. In numerical experiments, we compare our approach with traditional CNRM approximations. Our approach shows promising results in terms of speed of convergence and solution quality, and it almost always gives the best expected revenues.

5.1 Introduction and literature review

Selling perishable resources raises issues concerning what products to offer and when and to whom to sell them. It is not usually considered as a single optimization problem because of its intractability and because it involves both the marketing and sales departments. It is therefore divided into subproblems. The first subproblems forecasts the demand or the level of consumer interest. This is usually done by analyzing previous sales or by performing polls and studies. The second subproblem combines the forecast with marketing expertise to determine the products to offer. This includes the prices and the cancellation and reimbursement conditions. It remains to match the products to the demand in order to maximize revenue. This is done by controlling the product availability throughout the reservation period via a so-called *policy*. This revenue management (RM) is the subject of this article.

One of the first to propose an RM model was Littlewood (1972). He established a rule to protect the sales of different products for a single resource. RM was first applied by airlines. Researchers such as Talluri et van Ryzin (1998, 1999) and Bertsimas et Popescu (2003) then started to investigate RM with multiple resources. This allowed them to integrate underlying network effects such as *buy-across*, which refers to the choice a customer makes between two different but related products. The approach increased the complexity of the problem but gave more robust solutions with higher expected revenues.

The choice behavior logic has received much attention in the RM literature. Early on, Belobaba (1987b, 1989) explored it for the single-resource problem. Today there are both parametric models such as the multinomial logit (MNL), introduced by Ben-Akiva et Lerman (1985), and nonparametric models such as the preference list (PL), discussed by Rusmevichientong *et al.* (2006), Farias *et al.* (2013), van Ryzin et Vulcano (2015) and van Ryzin et Vulcano (2017). The choice of model usually depends on the practical context or the company history. Ideally, an RM model should be able to work with any choice behavior model. A choice behavior model can capture the *buy-up* effect, which occurs when a customer buys a more expensive product than originally planned. The solution is more robust if it takes into account the choices considered by the customers, and it almost always gives a higher expected revenue in practice. However, the additional complexity leads to longer computational times. Today RM routinely incorporates choice behavior, and the problem has become the choice network revenue management problem (CNRM). For a detailed review of RM, see chapter three of Talluri et van Ryzin (2004b).

The CNRM problem was introduced by Gallego *et al.* (2004), and Talluri et van Ryzin (2004a) presented an exact dynamic programming (DP) formulation. Because the DP rapidly becomes intractable, researchers have proposed various approximations, returning solutions that are either dynamic or static. The goal is to rapidly find the policy that returns the highest expected revenue in a simulation.

The most popular static approximation is the choice deterministic linear program (CDLP) of Liu et van Ryzin (2008). It indicates for how long each set of products, also called an offer, must be sold over the reservation period. This linear program can work with any choice behavior but has an exponential number of variables corresponding to every possible offer. Some researchers overcome this limitation by using column generation to solve the CDLP. To be effective the column generation subproblem must be specific to the choice behavior, which limits the applicability of the model. Bront *et al.* (2009) and Rusmevichientong *et al.* (2014) have proposed an exact formulation and a heuristic subproblem for the MNL model. For the PL model, Hosseinalifam *et al.* (2015) give an exact subproblem formula-

tion. Other researchers have focused on relaxing the CDLP via a decomposition by segment such as the segment-based deterministic concave program (SDCP) of Talluri (2010). The resulting problem is more tractable than the CDLP because the consideration sets are not large. However, it returns a weaker upper bound unless the segments overlap. To tighten the SDCP formulation, Meissner *et al.* (2013) add product constraints to link durations between segments, Talluri (2014) uses simulation, and Strauss et Talluri (2017) add consistency equalities. The sales-based linear program (SBLP) introduced by Gallego *et al.* (2011) is a compact formulation of the SDCP under MNL choice behavior. Reoptimization provides a way to address the dynamic aspect of the problem without solving the DP or considering a dynamic approximation.

Dynamic approximations often use the solutions of static approximations to decompose the DP formulation. For example, Liu et van Ryzin (2008) and Bront *et al.* (2009) use the dual prices of the CDLP resource constraints to evaluate the marginal value for each resource network and thus decompose the DP by resource. Similarly, Adelman (2007) uses an affine DP approximation to calculate dynamic bid prices. Erdelyi et Topaloglu (2010) vary the way in which the resource marginal values are weighted in the decomposition. Zhang (2011) present a nonlinear and nonseparable decomposition in which the subproblems exchange intermediate values. Finally, Zhang et Weatherford (2017) compare decomposition heuristics for the hotel industry and conclude that their performance is scenario-dependent.

Most static or dynamic approximations use the CDLP, which is solved rapidly with methods specific to the choice behavior model. We propose a new static approximation that is CDLP-independent and can work with any choice behavior. Our work is directly motivated by the work of blinded for peer review (2018), which returns a product closing (PC) policy determining when in the reservation period to close the sale of each product. There is no reopening of the sales, which differs from the policy given by dynamic decomposition. However, a no-reopening policy is sometimes mandatory, and the solution can serve as a good initial solution for an approximation that allows reopening.

5.2 Overview

The remainder of this paper is organized as follows. In Section 3 we introduce the notation of the CNRM problem, and a running instance (RI) is provided to support the explanations. We give the exact DP formulation and discuss its limitations; we then present static and dynamic approximations. In Section 4, we introduce the demand tree that represents choice behavior. We then apply the PC policy to the demand tree to determine for how long each product is sold. This is illustrated by a buying tree. Finally, by merging the behavior and

the offer, we shrink the buying tree and obtain a directed acyclic buying graph corresponding to our buying graph program (BGP). We then prove that any PC policy has a unique OP policy and show that the optimal BGP revenue is a lower bound for the CDLP.

In Section 5, we linearize the BGP by using hierarchy binaries to form the mixed buying graph program (MBGP). We also show that the use of hierarchy binaries has many advantages in practice; in particular, they accelerate the solution of the MBGP branch and bound. We also develop an iterative method for the BGP that becomes an effective heuristic for large instances. We explore various ways to select the buying-graph node to add at each iteration. In Section 6 we perform numerical experiments to compare our BGP approach to the principal CNRM approximations. The largest instance, corresponding to an airline network, clearly demonstrates the advantages of our approximation. Section 7 provides concluding remarks.

5.3 Model

We now introduce the notation and the RI. We then give the exact formulation of the CNRM problem and discuss the two types of approximations.

5.3.1 Notations

To illustrate the concepts, we use the RI shown in Figure 5.1. It can be interpreted as a transportation network with three cities. In the CRNM problem, we sell m resources $i \in I$ with capacity c_i. We offer these resources to the customers through n products $j \in J$ at price r_j. The incidence matrix $A = [a_{ij}]_{i \in I, j \in J}$ reflects whether a resource is used by a product ($a_{ij} = 1$) or not ($a_{ij} = 0$). Each segment $l \in L$ groups the customers with identical choice behavior and considering n_l products $C_l \subseteq J$. They buy a product according to a Poisson process with rate λ_l. Their behavior is modeled by the probability $P_l(j|S)$ of buying product j if S is the offer (set of products) proposed. The RI has two segments. The total arrival rate for a product with respect to an offer is $\lambda_j(S) = \sum_{l \in L} \lambda_l P_l(j|S)$.

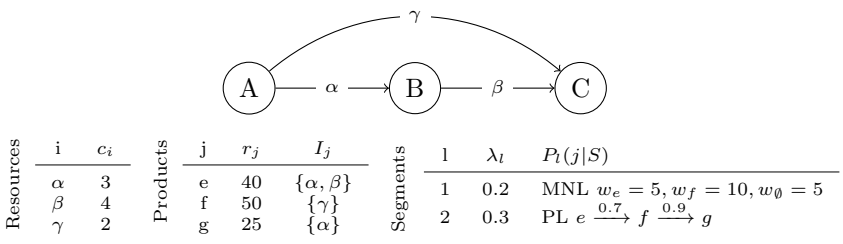

Figure 5.1 Running instance (RI).

We integrate two choice behaviors in the RI which are the multinomial logit (MNL) for segment 1 and the preference list (PL) for segment 2. Both are respectively representative of parametric and nonparametric choice behaviors. It allows us to show the flexibility of our approach regarding the choice behavior.

As explained in Ben-Akiva et Lerman (1985), the MNL probability function is

$$P_l(j|S) = \frac{w_l^j}{w_l^\emptyset + \sum\limits_{\gamma \in S} w_l^\gamma}, \quad \forall l \in L, j \in J, S \subseteq J. \tag{MNL}$$

Where a segment attributes weights w_l^j for each considered product and $w_l^\emptyset$ for quitting.

As presented in blinded for peer review (2018), the PL is defined by

$$P_l(j|S) = \begin{cases} \prod\limits_{k=1}^{l_j} \theta_l^k & \text{if } S \cap C_l^{l_j} = \{j\} \\ 0 & \text{otherwise.} \end{cases}, \quad \forall l \in L,\ j \in J,\ S \subseteq J. \tag{PL}$$

Where each product considered is a ranked choice l_j with probability transitions θ_l^k between consecutive choices.

5.3.2 Exact formulation

Each product can be sold throughout the reservation period, starting at $t = 0$ and finishing at $t = T$ when the resources perish. We select a sufficiently small time step h so that there is at most one customer arrival between t and $t + h$. The Bellman equations to maximize the value $V(t, x)$ of the remaining capacities x can then be written as follows:

$$\begin{aligned} V(t,x) = V(t+h,x) \\ + \max_{S \subseteq J(x)} \sum_{j \in S} \lambda_j(S) h \left(r_j - \Delta V_j(t+h,x)\right) \end{aligned} \tag{DP}$$

with the boundary conditions

$$\begin{aligned} V(t,0) = 0, \quad \forall t \in [0,T] \\ V(T+h,x) = 0, \quad \forall x \geq 0. \end{aligned}$$

where $J(x)$ is the set of products with available resources and $\Delta V_j(t,x) = V(t,x) - V\left(t, x - A_j\right)$ is the opportunity cost of selling product j at time t with A_j the incidence matrix column of product j. The solution provides an optimal dynamic policy specifying the offer $S^\star(t,x)$ to sell at each time t and the remaining capacities x in order to maximize revenue.

For the RI, we have $\sum_{l \in L} \lambda_l T = 0.5 \times 30 = 15$ customers arriving over the reservation period. The resource capacities form $\prod_{i \in I} c_i = 3 \times 4 \times 2 = 24$ combinations. To give an idea of the DP intractability, we note that there are $15 \times 24 = 360$ states for this small instance. The CNRM problem is therefore almost always solved approximately.

5.3.3 Approximations

The first type of approximation is called static because it considers the overall demand rather than individual arrivals. It is based on the following structure:

$$\begin{aligned} R = \quad & \max_{q} \; r^\top q \\ s.t. \quad & Aq \leq c, \quad (\pi), \\ & q \geq 0. \end{aligned} \tag{STATIC}$$

where $q = \{q_j\}_{j \in J}$ is the vector of product bookings under a certain demand and policy. The objective function maximizes the revenue, and the constraints ensure that the capacities are respected. An immediate policy is the product booking (PB) that sets the sales limit to the optimal $q_j^\star$ for each product:

$$S^{\text{PB}}(t, x) = \{j \in J \mid q_j \leq q_j^\star\}, \quad \forall t \in [0, T], x \leq c. \tag{PB policy}$$

This policy is static because it is fixed over the reservation period.

The most popular static approximation is the CDLP (Liu et van Ryzin, 2008), which is based on

$$\begin{aligned} q = \quad & \sum_{S \subseteq J} \lambda(S) d_S \\ s.t. \quad & \sum_{S \subseteq J} d_S \leq T, \\ & d_S \geq 0, \quad \forall S \subseteq J. \end{aligned} \tag{CDLP}$$

where $d_S \geq 0$ is the static policy indicating for how much time each offer should be available. By ordering this durations arbitrarily, we obtain the offer period (OP) policy defined as follow:

$$S^{\text{OP}}(t, x) = \{j \in J \mid j \in S, t \in [t_S, t_S + d_S]\}, \quad \forall t \in [0, T], x \leq c. \tag{OP policy}$$

Where t_S depends on how offers are ordered. We can solve the static approximation several times over the reservation period to obtain a more “dynamic" policy.

The second type of approximations calculates the pseudo-revenue $r_j - \Delta V_j(t+h, x)$ of each product without solving the entire DP. Most of these approaches implement a decomposition by resource to reduce the number of states. For example, Bront *et al.* (2009) approximate the network value function for resource i as:

$$V(t, x) \approx V_i(t, x_i) + \sum_{k \neq i} \pi_k^\star x_k \tag{DCOMP}$$

where the dual prices $\pi_k^\star$ come from the optimal solution of a static approximation. By substituting this expression into the DP we obtain one DP per resource for the calculation of $V_i(t, x)$. The opportunity cost $\Delta V_j(t, x)$ can then be calculated by an approximation based on $V_i(t, x)$, for example (Bront *et al.*, 2009):

$$\Delta V_j(t, x) \approx \Delta \widetilde{V}_j(t, x) = \sum_{\substack{i \in I \\ a_{ij}=1}} \frac{1}{m} \Delta V_i(t, x_i) + \frac{m-1}{m} \pi_i.$$

Where $\Delta V_i(t, x) = V_i(t, x_i) - V_i(t, x_i - 1)$. Other approximations have been proposed by Zhang et Weatherford (2017) and Erdelyi et Topaloglu (2010). An offer can then be determined for each arrival by solving:

$$S^{\mathrm{OD}}(t, x) = \arg \max_{S \subseteq J(x)} \sum_{j \in S} \lambda_j(S) h \left(r_j - \Delta \widetilde{V}_j(t+h, x) \right), \quad \forall t \in [0, T], x \leq c. \tag{OD policy}$$

The policy for the product availability over the reservation period is called the OD. This approach is dynamic because it changes depending on the arrivals.

5.4 Buying graph approximation

We now present our static approximation for the CNRM problem. It is inspired by the work of blinded for peer review (2018) in the sense that it uses and returns the same PC policy. The main difference is that our approximation is transformed to accept any choice behavior.

5.4.1 Demand tree

Any demand can be decomposed into buying paths u each representing the order in which customers are willing to buy n_u distinct products $j_u^k \in J$ with $k \in [1, n_u]$:

$$u = \left\langle j_u^1, \ldots, j_u^{n_u} \right\rangle, \quad \forall u \in U$$

Here $U = \{u \in \mathfrak{G}(S) \mid S \subseteq C_l,\, l \in L\}$ denotes the set of buying paths, where each is one of the permutations $\mathfrak{G}(S)$ of a subset S, and S is itself a subset of a segment consideration set C_l. We denote by $j_u = j_u^{n_u}$ the final product of any buying path. In other words, a buying path represents the customers who will buy j_u if it is offered and if all j_u^k with $k \in [1, n_u[$ are unavailable.

We define the operator $\oplus$ that add a buying path at the end of another. For two buying paths u_1 and u_2, we have $u_1 \oplus u_2 = \langle j_{u_1}^1, \ldots, j_{u_1}^{n_{u_1}}, j_{u_2}^1, \ldots, j_{u_2}^{n_{u_2}} \rangle$. Each buying path u, except the root $\langle\rangle$, has a unique parent $\text{par}_u = \langle j_u^1, \ldots, j_u^{n_u - 1} \rangle$, which is the buying path without its final product: $u = \text{par}_u \oplus \langle j_u \rangle$. We also define the unordered set S_u of products contained in the buying path u.

The probability $P_l(u)$ that segment l has customers following buying path u is by recurrence

$$P_l(u) = P_l(\text{par}_u) P_l\left(j_u | C_l \setminus S_{\text{par}_u}\right), \quad \forall u \in U, l \in L.$$

with $P_l(\langle\rangle) = 1$ and the second probability is defined for any choice behavior as seen is Section 5.3.1. The buying-path arrival rate vector $\lambda(u) = \{\lambda_j(u)\}_{j \in J}$ is obtained by summing the arrival probabilities of the segments: $\lambda_j(u) = \sum_{l \in L} \lambda_l P_l(u)$ if $j = j_u$ and 0 otherwise. Clearly, $\lambda(\text{par}_u) \geq \lambda(u)$. We can easily prove that:

$$\lambda_j(S) = \begin{cases} \sum\limits_{\substack{u \in \mathfrak{G}(O) \\ O \subseteq J \setminus S}} \lambda(u \oplus \langle j \rangle) & \text{if } j \in S \\ 0 & \text{otherwise} \end{cases}, \quad \forall j \in J, S \subseteq J. \tag{5.1}$$

For the RI, if $j = e$ and $S = \{e, f\}$ then $U = \{g\}$ so that $\lambda_e(\{e, f\}) = \lambda(\langle e \rangle) + \lambda(\langle g, e \rangle)$.

Because the parent is unique, the buying paths form a tree called the demand tree where each node is a unique buying path u with an incoming arc weighted $\lambda(u)$ from its parent par_u. Figure 5.2 shows the demand tree of the RI. For example, node f after S and e corresponds

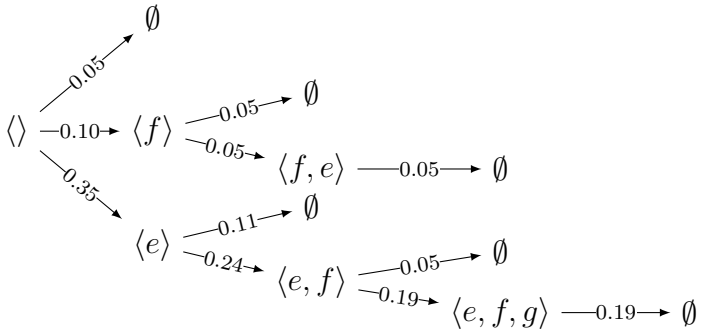

Figure 5.2 Demand tree for the RI.

to the buying path $\langle e, f\rangle$, which represents customers who buy f if e is not available. These customers arrive at the rate $\lambda(\langle e, f\rangle) = 0.24$. Each $\emptyset$ node indicates customers who leave the reservation system without making a purchase.

This demand tree can represent any choice behavior. We explain in the next section how we use it to build our CNRM approximation.

5.4.2 Buying graph

We base our approximation on the PC policy, as in blinded for peer review (2018), which ends the sale of each product at time $t_j \in [0, T]$.

$$S^{\mathrm{PC}}(t, x) = \{j \in J \mid t \leq t_j\}, \quad \forall t \in [0, T], x \leq c. \tag{PC policy}$$

A buying path is active if its customers are effectively buying the final product, j_u. Under the PC policy, it is active if this PC t_{j_u} is greater than every other PC t_j with $j \in \mathrm{par}_u$ of the buying path. Let d_u be the duration of the activity. Then

$$d_u = \left(t_{j_u} - \max_{j \in \mathrm{par}_u} t_j\right)^+, \quad \forall u \in U.$$

We represent the buying-path durations in a buying logic tree. The input arc of each node is the time for which the buying path is active for the given PC policy. The buying logic tree for the RI is illustrated in Figure 5.3. For example, the buying path $\langle e, f, g\rangle$ is active if t_f is

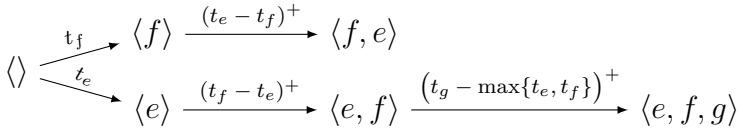

Figure 5.3 Buying logic tree for the RI.

greater than the maximum of t_e and t_f.

Let $t_u = \max_{j \in u} t_j$ be the buying-path closing time. Then

$$d_u = t_u - t_{\mathrm{par}_u}, \quad \forall u \in U.$$

The maximum function is independent of the order of the elements, so if we let $t_u = t_{S_u} = \max_{j \in S_u} t_j$ be the offer closing, we have

$$d_u = t_{S_u} - t_{S_{\mathrm{par}_u}}, \quad \forall u \in U.$$

Therefore, the complexity is based on combinations rather than permutations. In fact all the products sets S_u generated by all the buying paths u belongs to the set

$$C_L = \bigcup_{l \in L} \{S_u \mid P_l(u) \neq 0,\, u \in \mathfrak{G}(S),\, S \subseteq C_l,\, l \in L\}$$

This is the union of consideration subsets that have at least one corresponding buying path nonnull. The cardinality $|C_L|$ is in practice much lower than $|U|$ and our approximation benefits from overlapping. A way to determine C_L is to enumerate all the buying paths with nonnull probabilities and to obtain the subsets by merging the buying paths that share exactly the same products. For the RI, the first segment has five nonnull buying paths, $\langle\rangle$, $\langle e\rangle$, $\langle f\rangle$, $\langle e, f\rangle$, and $\langle f, e\rangle$. By merging we obtain four subsets: $\{\}, \{e\}, \{f\}, \{e, f\}$. The second segment has four nonnull buying paths, $\langle\rangle$, $\langle e\rangle$, $\langle e, f\rangle$, and $\langle e, f, g\rangle$. By merging we obtain four subsets: $\{\}, \{e\}, \{e, f\}, \{e, f, g\}$. The union of the two segments gives $C_l = \{\}, \{e\}, \{f\}, \{e, f\}, \{e, f, g\}$.

We can represent the buying logic as a buying graph with nodes corresponding to the PC t_{S_u} and arcs to the arrival rate vector $\lambda(u)$ from various subset of S_u. Figure 5.4 shows the buying graph for the RI.. This graph is directed and acyclic. In other words, the arrival rate

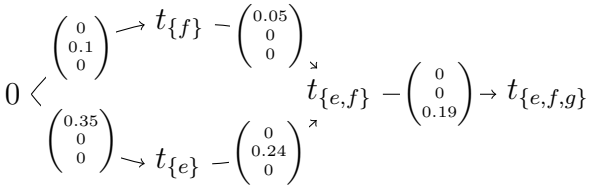

Figure 5.4 Buying graph for the RI.

vector on each arc is active during the time between the origin and destination nodes, which are PC times.

When a buying path is active, the only product sold is j_u. It is sold at a rate corresponding to the total arrival rate $\lambda(u)$ of the buying path. With the duration d_u, we can finally formulate our BGP based on the STATIC approximation of Section 5.3.3:

$$\begin{aligned} q &= \sum_{u \in U} \lambda(u) d_u && && \text{(BGP)} \\ s.t.\quad & d_u = t_{S_u} - t_{S_{\mathrm{par}_u}}, && \forall u \in U, \\ & t_S = \max_{j \in S}\ t_j, && \forall S \in C_L, \\ & t_j \geq 0, && \forall j \in J. \end{aligned}$$

The complexity of our approximation lies in the $|C_L|$ non linear constraints that are based on the n product closing times. This number is at most $\sum_{l \in L} 2^{n_l} - 1$ when there is no

overlapping and can be reduced considerably when segments overlap. It is much less than the CDLP $2^n - 1$ variables by reasonably assuming that customers do not consider buying all products (n_l much smaller than n).

5.4.3 Quality of the BGP

Our approximation uses a PC policy that we first compare to the OP policy.

Proposition 6. *A PC policy always has a unique equivalent OP policy, denoted OP_{PC}.*

Proof. Proof of Proposition 6: The PCs $\{0 \leq t_j \leq T\}_{j \in J}$ are ordered by time to give n periods; some durations will be null if some products close simultaneously. $\mathrm{OP_{PC}}$ is then the periods indexed by $k \in [1, n]$, starting at t_{k-1} ($t_0 = 0$), finishing at t_k, and offering $S_k = \{j \mid t_j > t_k\}$. The proof of the uniqueness is direct. □

The previous equivalence allows us to conclude on the relation between CDLP and BGP in terms of revenue.

Proposition 7. $R_{BGP}(PC) = R_{CDLP}(OP_{PC})$.

Proof. Proof of Proposition 7: We use the notations defined in the proof of proposition 6. The CDLP product quantity vector for $\mathrm{OP_{PC}}$ is then:

$$q_j^{\mathrm{CDLP}}(\mathrm{OP_{PC}}) = \sum_{S \subseteq J}^{n} \lambda_j(S) d_S = \sum_{k=1}^{n} \lambda_j(S_k) d_{S_k}$$

With Equation 5.1, we can write:

$$q_j^{\mathrm{CDLP}}(\mathrm{OP_{PC}}) = \sum_{\substack{k=1 \\ j \in S_k}}^{n} \sum_{\substack{u \in \mathfrak{G}(O) \\ O \subseteq J \setminus S_k}} \lambda(u \oplus \langle j \rangle) d_{S_k}$$

By enumerating using the logical function $1\left\{S_{\mathrm{par}_u} \subseteq J \setminus S_k\right\}$ and inverting the sums we obtain:

$$q_j^{\mathrm{CDLP}}(\mathrm{OP_{PC}}) = \sum_{\substack{u \in U \\ j_u = j}} \sum_{\substack{k=1 \\ j \in S_k}}^{n} 1\left\{S_{\mathrm{par}_u} \subseteq J \setminus S_k\right\} \lambda(u) d_{S_k}$$

The condition $S_{\mathrm{par}_u} \subseteq J \setminus S_k$ implies that each product in the buying path par_u must be unavailable such that $S_{\mathrm{par}_u} \cap S_k = \emptyset$. It occurs after $t_{S_{\mathrm{par}_u}} = t_{s_u}$ with $s_u \in [1, n]$. The

condition $j \in S_k$ is equivalent to consider $t_{S_u} = t_{e_u}$ with $e_u \geq s_u$. Then

$$\begin{aligned} q_j^{\text{CDLP}}\left(\text{OP}_{\text{PC}}\right) &= \sum_{\substack{u \in U \\ j_u = j}} \lambda(u)(t_{e_u} - t_{e_u - 1} + t_{e_u - 1} + ... - t_{s_u}) \\ &= \sum_{\substack{u \in G \\ j_u = j}} \lambda(u) \left(t_{S_u} - t_{S_{\text{par}_u}}\right) = q_j^{\text{BGP}}(PC) \end{aligned}$$

We have $\sum\limits_{S \subset J} d_S = \sum\limits_{k=1}^{n} d_{S_k} \leq T$ for feasibility. Therefore, $R_{\text{BGP}}\left(\text{PC}\right) = R_{\text{CDLP}}(\text{OP}_{\text{PC}})$. □

We can directly derive the next proposition.

Proposition 8. $R^{\star}_{\text{BGP}} \leq R^{\star}_{\text{CDLP}}$.

Proof. Proof of Proposition 8: Let PC^* be the optimal policy of the BGP. It has an equivalent OP_{PC^*} such that $R_{\text{BGP}}\left(\text{PC}^*\right) = R_{\text{CDLP}}(\text{OP}_{\text{PC}^*})$ by Proposition 6. OP_{PC^*} is not necessarily the optimal CDLP solution. □

However, OP does not always have a PC equivalent. A period offering $\{f\}$, another offering $\{e\}$, and a third offering $\{e, f\}$ is a valid OP policy that has no corresponding PC policy. In fact, the PC policy enforces no-reopening, whereas the OP policy is able to propose f again in our example above. The no-reopening condition is formulated as follows:

$$\textit{no-reopening} \Leftrightarrow \forall j \in S_t \text{ then } \forall \theta < t \;\; j \in S_\theta$$

A no-reopening policy could be mandatory in practice for social or marketing reasons. Adding no-reopening constraints to OP-based approximations complicates the models, whereas our approximation naturally ensures no reopening.

Because of no reopening, the optimal DP revenue could be higher than that for the BGP. However, if we enforce no-reopening we can prove the next proposition.

Proposition 9. *If no-reopening, an OP policy always has a unique equivalent PC policy, denoted PC_{OP}, and the BGP is equivalent to CDLP.*

Proof. Proof of Proposition 9: Without reopening, the periods can be ordered only by PC times. Similarly to the proof of proposition 6, we can prove that $R_{\text{CDLP}}\left(\text{OP}\right) = R_{\text{BGP}}(\text{PC}_{\text{OP}})$. Combining the two propositions gives $R_{\text{CDLP}}\left(\text{OP}^*\right) = R_{\text{BGP}}(\text{PC}_{\text{OP}^*}) = R_{\text{BGP}}\left(\text{PC}^*\right) = R_{\text{CDLP}}(\text{OP}_{\text{PC}^*})$. □

Moreover, if there is no reopening the BGP inherits the features of CDLP found by Liu et van Ryzin (2008), in particular:

— BGP is asymptotically optimal, which ensures that the revenue converges to the optimal DP revenue when the capacity and demand are scaled up proportionally.

— $R^\star_{\text{DP}} \leq R^\star_{\text{BGP}}$.

Proposition 6 implies that any solution of the BGP can be given to the CDLP. The CDLP is usually solved by column generation, which can use any OP policy for the initial columns. We denote by CDBG the following program:

$$\begin{array}{ll} \text{CDLP} & \text{by column generation} \\ s.t. & \text{Initial columns given by the BGP} \end{array} \tag{CDBG}$$

This program allows us to re-open any PC policy from the BGP by starting the CDLP solution process with OP_{PC}. We are also sure to have $R_{\text{CDBG}}(OP^*) = R_{\text{CDLP}}(OP^*) \geq R_{\text{CDBG}}(\text{OP}_{\text{PC}}) = R_{\text{BGP}}(\text{PC})$ for OP^*, the optimal CDBG solution.

5.5 Linearization and iterative resolution

We now explain the methods we use to solve our approximation effectively even for large instances. Section 5.5.1 generalizes the linearization developed by blinded for peer review (2018) for the nonparametric choice behavior model. Section 5.5.2 introduces a new method to solve our approximation on a buying graph revealed progressively according to building strategies detailed in Section 5.5.3.

5.5.1 Linearization with hierarchy

The BGP nonlinearity is caused by the constraint $t_S = \max_{j \in S} t_j$, which appears $|C_L|$ times. To remove this nonlinearity, we use binary variables. They are called hierarchy variables because they rank products by PC time:

$$h^j_{j'} = \begin{cases} 1 & \text{if } t_j > t_{j'} \\ 0 & \text{otherwise} \end{cases} \quad , \quad \forall j, j' \in J.$$

Clearly, $h^{j'}_j = 1 - h^j_{j'}$.

Any set $S \in C_L$ corresponds at least to one buying path u ($S = S_u$), which has a parent par_u. The parent set S_{par_u} is a subset of S_u with $S_u \setminus S_{\text{par}_u} = \{j_u\}$. The set S always has at least another subset $S_v \subset S$ corresponding to a buying path $v \neq u$ such that $j_u \in v$ (for example, $\{j_u\}$). Because $S_{\text{par}_u} \cup S_v = S_u = S$, we have

$$t_S = \max\{t_{S_{\text{par}_u}}, t_{S_v}\}$$

The first consequence is that $t_S \geq t_{S_{\text{par}_u}}$ and $t_S \geq t_{S_v}$. We have $S_v \setminus \{j_u\} \subset S_{\text{par}_u}$. Therefore, if j_u is superior in hierarchy than every products of S_{par_u} not contained in S_v then $t_S = t_{S_v}$. So that

$$t_S \leq t_{S_{\text{par}_u}} + h_j^{j_u} T, \quad \forall j \in S \setminus S_v.$$

Otherwise if one product of S_{par_u} is superior in hierarchy than j_u then $t_S = t_{S_{\text{par}_u}}$. We thus have

$$t_S \leq t_{S_v} + \sum_{j \in S \setminus S_v} h_{j_u}^j T$$

For the set $\{e, f, g\}$ of the RI, the parent subset is $\{e, f\}$ so that $j_u = g$. We assume that $\{e, g\}$ is the other subset "S_v". Then, $t_{\{e,f,g\}} = t_{\{e,g\}}$ if $h_f^g = 1$ and $t_{\{e,f,g\}} = t_{\{e,f\}}$ otherwise. The previous linearization is $t_{\{e,g\}} \leq t_{\{e,f,g\}} \leq t_{\{e,g\}} + h_g^f T$ and $t_{\{e,f\}} \leq t_{\{e,f,g\}} \leq t_{\{e,f\}} + h_f^g T$.

We finally obtain the MBGP as follows:

$$\begin{aligned}
q = & \sum_{u \in U} \lambda(u) d_u && && \text{(MBGP)} \\
s.t. \quad & d_u = t_{S_u} - t_{S_{\text{par}_u}}, && && \forall u \in U, \\
& S_{\text{par}_u} \cup S_v = S, && \exists S_v \in C_L, && \\
& t_S \geq t_{S_{\text{par}_u}}, && && \\
& t_S \geq t_v, && && \forall S \in C_L, \\
& t_S \leq t_{S_{\text{par}_u}} + h_j^{j_u} T, && \forall j \in S \setminus S_v, && \\
& t_S \leq t_{S_v} + \sum_{j \in S \setminus S_v} h_{j_u}^j T, && && \\
& h_{j'}^j \in \{0, 1\}, && \forall j, j' \in J. &&
\end{aligned}$$

To limit the number of constraints, we must find for each set S the subset S_v with the highest cardinality in order to reduce the size of the set $S \setminus S_v$. Contrarily to a majority of approximations, we thus benefit from overlapping because it produces bigger subsets S_v.

We could have linearized the BGP with possibly fewer constraints by using a binary b_{S_u} to model the closing order between set S_{par_u} and product j_u, obtaining: $t_{S_u} \geq t_{S_{\text{par}_u}}$, $t_{S_u} \geq t_{j_u}$, $t_{S_u} \leq t_{S_{\text{par}_u}} + b_{S_u} T$, and $t_{S_u} \leq t_{j_u} + (1 - b_{S_u}) T$. However, the use of hierarchy binaries is more appropriate for several reasons:

— In practice, companies often know at least partially the PC order. We can use this knowledge by fixing binary variables, reducing the solution time. If the order is fully known, every binary variable can be fixed and our approximation becomes the linear

buying-graph program (LBGP):

$$
\begin{aligned}
q = & \sum_{u \in U} \lambda(u) d_u && && \text{(LBGP)} \\
s.t. \quad & d_u = t_{S_u} - t_{S_{\text{par}_u}} && && \forall u \in U. \\
& t_S = t_j,\ j \in S \mid \widehat{h}^j_{j'} = 1, \forall j' \in S, && && \forall S \in C_L. \\
& \widehat{h}^j_{j'} \in \{0,1\} \text{ is fixed} && && \forall j, j' \in J \\
& t_j \geq 0, && && \forall j \in J
\end{aligned}
$$

— We can intuitively propose good hierarchies. For example, we can rank products by price. This leads to good initial solution(s) for the branch and bound, making the problem easier to solve.

— We can solve the MBGP offline and use the corresponding optimal hierarchy to solve the LBGP in the time allowed.

5.5.2 Iterative resolution

The buying graph may be large, and it directly impacts the number of binaries and constraints in the MBGP. The worst choice behavior for our approximation is the MNL because it fully expands the tree for each segment consideration set. For example, an MNL segment with ten products ($n_l = 10$) leads to $\lfloor e \times n_l! \rfloor \approx 10^7$ buying paths. Our PC formulation has only $2^{n_l} - 1 = 1023$ variables, and our formulation with hierarchy binaries has $\frac{n_l(n_l-1)}{2} = 45$ binaries. The advantage of the MNL is that each offer S has at least two distinct subsets of cardinality $|S|-1$. We thus need four constraints to linearize each PC, which is the minimum we can achieve. The MBGP for this consideration set therefore has m capacity constraints and $4 \times (1023 - n_l) + n_l = 4062$ constraints to model the choice behavior. In this section we present a method for the BGP and inherited approximations that progressively adds nodes to the buying graph. Let $\widetilde{BGP}$ be the BGP solved on the set $\widetilde{U} \subseteq U$:

$$
\begin{aligned}
q = & \sum_{u \in \widetilde{U}} \lambda(u) d_u && && (\widetilde{BGP}) \\
s.t. \quad & d_u = t_{S_u} - t_{S_{\text{par}_u}}, && && \forall u \in \widetilde{U}, \widetilde{U} \subseteq U \\
& t_S = \max_{j \in S} t_j, && && \forall S \in \widetilde{C_L}, \\
& t_j \geq 0, && && \forall j \in J.
\end{aligned}
$$

We can easily prove that it converges to the BGP revenue. The buying graph constructed must have a good representation of the demand so that we can rapidly obtain a good PC policy for the full demand.

5.5.3 Progressive buying graph

To follow the progressive building of any buying graph $\widetilde{U} \subseteq U$, we introduce the following indicators:

Arrived is the percentage of customers arrived in $\widetilde{U}$:

$$\text{Arrived}(\widetilde{U}) = \frac{1}{\lambda(\langle\rangle)} \sum_{u \in \widetilde{U} \,|\, \text{par}_u = \langle\rangle} \lambda(u)$$

It is 100% when all first layer nodes (one product buying path) have been added.

Revealed is the percentage of customers completely known in terms of choice behavior. It therefore corresponds to the sum of quit nodes ($\emptyset$ in Figure 5.2) in $\widetilde{U}$:

$$\text{Revealed}(\widetilde{U}) = \frac{1}{\lambda(\langle\rangle)} \sum_{u \in \widetilde{U} \,|\, j_V = \emptyset} \lambda(u)$$

The 100% is reached only when $\widetilde{U} = U$.

Potential corresponds to the highest arrival ratio of nodes still not added:

$$\text{Potential}(\widetilde{U}) = \frac{1}{\lambda(\langle\rangle)} \max_{u \in U \setminus \widetilde{U}} \lambda(u)$$

As the construction, this indicator decreases.

The main challenge is to build progressively the buying graph with the objective to capture most of the behavior as soon as possible. In this vein, we add nodes by decreasing arrival ratio (AR) or by decreasing product of arrival ratio and price (ARP). The refinement of the demand has thus less and less impact as building the graph.

5.6 Numerical experiments

The following approximations are benchmarked:

CDLP (Liu et van Ryzin, 2008): Described in Section 5.3.3 and solved by column generation with the heuristic and exact subproblem of Bront *et al.* (2009) for MNL.

LBGP: Presented in Equation LBGP of Section 5.5.1. The hierarchy is given by the products price as: the more expensive, the higher in hierarchy.

MBGP: Presented in Equation MBGP of Section 5.5.1. The previous LBGP is used as initial solution in the branch and bound. The relative integrability gap is set to 10^{-3}.

$\widetilde{\text{MBGP}}_{\text{AR or ARP}}$: Presented in Equation $\widetilde{BGP}$ and linearized in Section 5.5.2. Nodes are added by decreasing arrival ratio (AR) or arrival ratio × price (ARP).

CDBG: The CDLP approximation with an initial solution given by the MBGP with a relative integrability gap of 10^{-2} as explained in Section 5.4.3.

We use the following policies:

OP is the OP policy described in Section 5.3.3. It is obtained by a lexicographic sequencing of the CDLP durations.

PB is the PB policy of static limit $q_j^\star$ for each product, as explained in Section 5.3.3.

PC is the PC policy returned by the PCP, as explained in Section 5.4.2.

OD is the OD policy described in Section 5.3.3. It is obtained by the dynamic decomposition of Bront *et al.* (2009).

The quality of an approximation depends on its solution time and the expected revenue. We use a Monte-Carlo approach on a discrete-arrival simulation to determine the expected revenue. We generate random discrete arrivals by generating arrival timings according to a Poisson process for each segment. Each simulation is stopped after a number of evaluations specific to the instance.

We conduct numerical experiments only on instances with the MNL choice behavior. First because it is one of the most difficult choice behavior for our approximation. Each consideration set results in a complete demand tree contrarily to PL choice behavior for example. Also because other approximations compared do not tackle more than one choice behavior simultaneously. And they have been mostly developed for the MNL.

We build scenarios by varying the quit percent $w_\emptyset^\%$ and the load factor LF. The quit ratio allows us to calculate the MNL quit weight for each segment as follows: $w_\emptyset = w_\emptyset^\% \times \sum_{j \in C_l} w_j^l$. The load factor is simply the sum of the arrivals over the sum of the capacities: $LF = \sum_{l \in L} \lambda_l / \sum_{i \in I} c_i$. By multiplying all the λ_l by the same factor, we obtain the desired LF.

5.6.1 Parallel flights

Our first instance, parallel flights, is illustrated in Figure 5.5. It is composed of three parallel flights, of capacity 100, from city A to city B at 09:00, 11:00, and 20:00. We consider two fares H (150) and L (100) per flight, giving six products. The reservation period is 360 periods. The customers are divided into four segments, as shown in Table 5.1. A complete description of this instance is given in the Section B.1 of the appendices.

We present in Table 5.2 the expected revenue and running time of the parallel flights instance for LF $\in \{0.8, 1.0, 1.2\}$ and $w_\emptyset^\% \in \{1, 5, 10\}$. The expected revenue E[R] is given as a relative

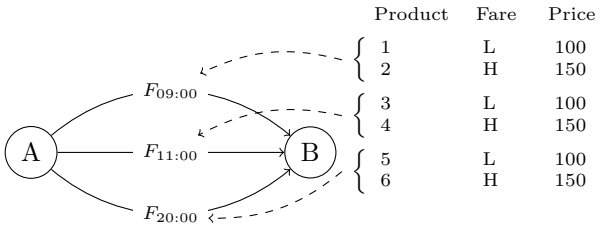

Figure 5.5 Resources and products for parallel flights .

Table 5.1 Segments for parallel flights with $w_\emptyset = 0.1$.

Segment	Arrival ratio	Choice behavior (MNL)
1	0.17	(1,9.1)(2,8.1)(∅,1.7)
2	0.25	(1,10.0)(3,10.0)(5,10.0)(2,8.9)(4,8.9)(6,8.9)(∅,5.7)
3	0.17	(3,10.0)(5,10.0)(4,8.9)(6,8.9)(∅,3.8)
4	0.25	(5,10.5)(6,9.3)(3,8.1)(4,7.2)(∅,3.5)

difference with CDLP-OP. The time CPU includes the approximation solving and policy building. The simulation evaluates the different policies in 2000 evaluations.

Table 5.2 Running seconds CPU and expected revenue E[R] for the Parallel flights instance

			CDLP			MBGP			CDBG			LBGP		
LF	$w_\emptyset^{\%}$		OP	PB	OD	PC	PB	OD	OP	PB	OD	PC	PB	OD
0.8	1	CPU	0.25	0.25	9.64	0.14	0.14	10.51	0.08	0.08	10.90	0.04	0.04	10.82
		E[R]	35267.9±0.29%	-2.98	-0.29	0.07	-2.91	0.01	-0.13	-3.04	0.14	-0.11	-2.97	0.01
	5	CPU	0.00	0.00	10.84	0.03	0.03	11.11	0.05	0.05	10.86	0.03	0.03	10.92
		E[R]	32538.2±0.30%	-3.13	-0.18	0.11	-2.83	-0.04	-0.05	-3.15	-0.13	-0.08	-3.02	-0.09
	10	CPU	0.02	0.02	10.90	0.08	0.08	11.79	0.06	0.06	11.67	0.02	0.02	11.92
		E[R]	29620.7±0.32%	-3.06	-0.00	0.36	-2.91	0.35	0.41	-3.12	0.30	0.18	-2.99	0.27
1.0	1	CPU	0.49	0.49	16.35	0.10	0.10	15.67	0.21	0.19	15.88	0.02	0.02	15.05
		E[R]	42764.6±0.17%	-0.08	0.60	-0.33	-0.42	-0.13	-0.14	-0.08	0.50	-5.76	-4.92	0.28
	5	CPU	0.25	0.25	14.45	0.10	0.10	14.28	0.08	0.08	14.38	0.02	0.02	14.18
		E[R]	40359.1±0.24%	-2.05	-0.05	0.07	-2.20	0.21	-0.07	-2.18	0.00	-0.10	-2.16	-2.04
	10	CPU	0.00	0.00	14.35	0.02	0.03	14.41	0.03	0.03	14.67	0.02	0.02	14.85
		E[R]	36984.0±0.27%	-2.95	-0.02	-0.26	-2.73	0.12	0.26	-2.60	0.01	0.23	-2.64	0.11
1.2	1	CPU	0.00	0.00	18.72	0.10	0.10	18.87	0.11	0.11	18.54	0.03	0.03	19.23
		E[R]	43231.8±0.15%	4.01	3.35	0.70	3.66	1.88	0.71	4.01	2.75	-6.59	-5.16	3.55
	5	CPU	0.02	0.02	18.09	0.09	0.09	17.75	0.17	0.17	17.74	0.02	0.02	17.89
		E[R]	43337.0±0.16%	2.85	3.61	0.24	2.92	3.49	0.06	2.94	3.43	-6.96	-5.77	1.42
	10	CPU	0.16	0.16	17.72	0.03	0.03	17.51	0.15	0.15	17.62	0.03	0.03	17.61
		E[R]	42247.3±0.16%	0.79	1.51	0.07	0.72	1.37	0.12	0.68	1.56	-4.73	-3.82	-2.48
$\overline{\text{CPU}}$			0.13	0.13	14.56	0.08	0.08	14.65	0.10	0.10	14.70	0.02	0.02	14.72
$\overline{\text{E[R]}}$			–	-0.73	0.94	0.11	-0.75	0.81	0.13	-0.73	0.95	-2.66	-3.72	0.11

We note that the approximations CDLP, MBGP, and CDBG give similar results respectively for the OD and PB control. At Table B.1 of the appendices, we note that the only scenarios for which they return different ideal revenues are $LF = 1.0$ with $w_\emptyset^{\%} = \{1, 5\}$. In these cases

CDLP and CDBG return a higher ideal revenue than MBGP surely because they both allow reopening. Otherwise, they share the same ideal revenue and capacity factor so the policies derived and the expected revenues are similar.

We observe that the OD policy is generating the best expected revenue. In average, it is 0.9% better than the CDLP-OP because its dynamic structure adapts to how customers arrive. However, it has the longest running time by far: 15 seconds versus some milliseconds. This is mainly due to the time for building the policy as we can see in Table B.2. In fact, for each resource we have to solve an optimization problem per remaining capacity and by future arrivals as seen in Section 5.3.3. The latter problem is small but it has to be solved more than 1000 times for each resource.

We also observe the unequal performance of the PB policy over the different scenario. It performs better as the load factor increases and as the quit ratio decreases. In our experiments, it passes from -3% for $LF = 0.8$ and $w_\emptyset^\% = 10$ to $+4\%$ for $LF = 1.2$ and $w_\emptyset^\% = 1$. Globally, the PB policy performs well when the load factor is higher than one. In fact, the PB policy, obtained via a static approximation, cannot capture more demand than predicted. It explains why the capacity factors in Table B.2 of the appendices are the lowest when the load factor is less than one.

We observe that the LBGP is the fastest approximation but returns the poorest policies. Its PC policy is 2.7% lower than the CDLP while the MBGP is 0.1% better. The insignificant difference in running time is not enough to overshadow the loss in revenue for this instance.

5.6.2 Bus-line

The bus-line instance has eight buses leaving at different hours from city A to cities B, C, D and E. Each bus serves ten markets, as illustrated in Figure 5.6. Each bus has a capacity

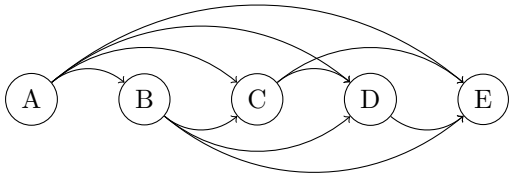

Figure 5.6 Markets of the bus-line instance.

of 30 and there are $8 \times 4 = 32$ resources. Two fares (low and high) are offered for each trip, giving a total of $10 \times 2 \times 8 = 160$ products. In the bus industry, tickets are usually available at least two months in advance, so we set $T = 60$ days. We generate eight segments by bus each considering between 2 and 6 products with an average of 4.8. In total there are

$8 \times 10 = 80$ segments. A complete description of the instance is given in the Section B.2 of the appendices.

The Bus-line instance is significantly larger than the parallel flights. We thus start by investigating the time to build the OD policy which is already very long for the parallel flight. It is directly related to the number of values N_{OD} to calculate and store $N_{\text{OD}} = \sum_{i \in I} c_i \times \sum_{l \in L} \lambda_l T$ corresponding to one value per resource, remaining capacity and arrival. For the bus-line instance $N_{\text{OD}} = (4N_b) \times (30) \times (30N_b) = 3600{N_b}^2$ with N_b the number of buses considered and $LF = 1$. Computing these values can be done off-line but becomes rapidly too computationally demanding for larger instances because there is a NP-hard problem to solve for determining each value. (see Figure B.1 of the appendices for examples of running times) That is why we now only compare the OP, PB and PC policies.

Table 5.3 Running seconds CPU and expected revenue E[R] for the Bus-line instance.

			CDLP		MBGP		CDBG		LBGP	
LF	$w_\emptyset^\%$		OP	PB	PC	PB	OP	PB	PC	PB
0.8	1	CPU	65.19	65.19	94.57	94.58	44.30	44.30	1.97	1.98
		E[R]	66510.5±0.17%	-3.37	0.21	-6.09	0.21	-3.53	-5.77	-9.06
	5	CPU	6.77	6.79	3.83	3.84	10.21	10.22	2.09	2.11
		E[R]	62074.4±0.19%	-2.83	0.25	-4.96	0.10	-3.27	-1.39	-5.41
	10	CPU	7.71	7.72	1.38	1.38	6.56	6.58	1.43	1.43
		E[R]	57175.2±0.19%	-2.66	0.63	-5.49	0.62	-3.09	0.77	-5.37
1	1	CPU	35.47	35.48	2564.61	2564.61	271.54	271.55	1.83	1.83
		E[R]	74183.4±0.14%	-0.91	-0.31	-3.65	-0.11	-1.01	-6.21	-9.15
	5	CPU	6.93	6.94	10.86	10.88	13.08	13.08	1.78	1.80
		E[R]	70299.7±0.15%	-1.64	0.55	-2.62	0.41	-1.20	-6.23	-9.45
	10	CPU	7.90	7.91	1.53	1.53	7.85	7.88	1.86	1.86
		E[R]	66462.7±0.16%	-2.44	0.18	-4.07	0.10	-2.12	0.28	-4.11
1.2	1	CPU	8.82	8.83	11424.59	11424.59	5020.03	5020.05	12.58	12.58
		E[R]	79578.9±0.13%	-0.07	-0.29	-2.36	-0.45	-0.04	-5.87	-6.22
	5	CPU	11.41	11.43	367.27	367.28	72.38	72.39	2.41	2.41
		E[R]	76215.1±0.13%	-0.16	0.32	-3.67	0.29	-0.45	-5.99	-8.20
	10	CPU	5.83	5.86	9.78	9.79	5.90	5.92	2.25	2.25
		E[R]	72399.0±0.14%	-0.92	0.12	-2.55	0.15	-0.63	-5.83	-7.21
$\overline{\text{CPU}}$			17.34	17.35	1608.71	1608.72	605.76	605.77	3.13	3.14
$\overline{\text{E[R]}}$			–	-1.67	0.18	-3.94	0.15	-1.71	-4.03	-7.13

We report at Table 5.3 the running time and expected revenue for the Bus-line instance for the load factor $LF \in \{0.8, 1.0, 1.2\}$ and $w_\emptyset^\% \in \{1, 5, 10\}$. Section B.2 of the appendices reports additional measures. The simulation used to evaluate policies has 2000 evaluations.

We observe the same tendency for the PB policy as for the previous instance. It better performs as the load factor increases and the quit ratio decreases passing from -3.4% at LF $= 0.8$ to -0.07% at LF $= 1.2$ and $w_\emptyset^\% = 1$. The same explanation occurs here.

We observe that the PC policy from the MBGP is in average 0.2% better than other policies. It could be explained by the precision of the simulation or the possible better structure of the PC policy. In fact, closing each product sales only once over the reservation period seems more robust than reopening product sales at different moments in the reservation period.

We note that CDBG-OP returns a slightly better expected revenue than the CDLP-OP. However they should be equals because both approximations are equivalent as we can see in Table B.3 of the appendices. This difference could be explained by the simulation precision or by a degenerate solution because of the initial PC solution.

In this example, the LBGP approximation is in average 6 times faster than others but returns a policy 4% worse than the CDLP-OP. As for the previous instance, the running time does not justify the loss in revenue. However, we start to see its potential for larger instances.

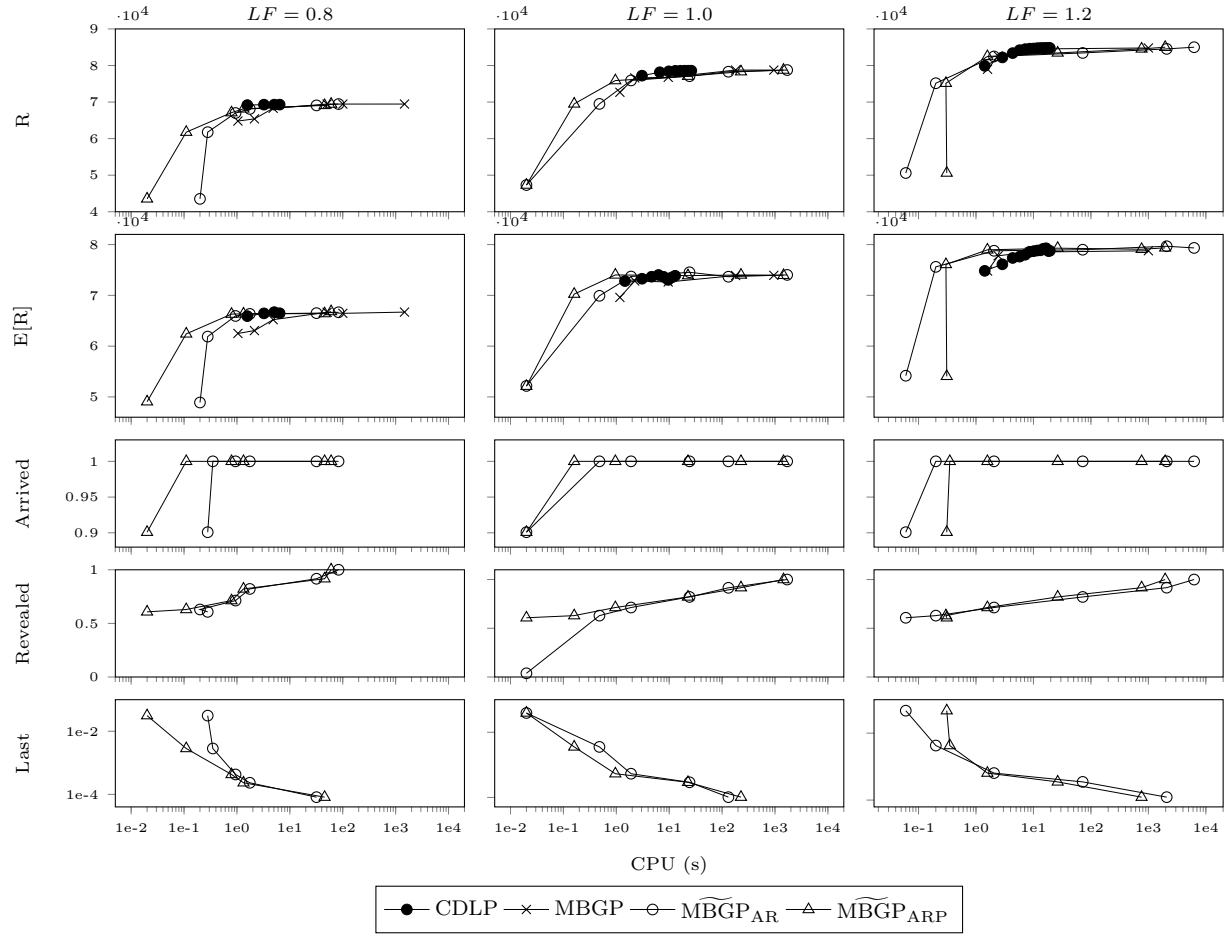

Figure 5.7 Convergence of different approximations for the Bus-line instance with $w_{\emptyset}^{\%} = 1$.

Note that the MBGP is very slow for the quit ratio $w_{\emptyset}^{\%} = 1$. We further investigate this issue by plotting the CDLP column generation and the MBGP branch and bound at Figure 5.7.

It shows that the MBGP converges almost as quickly as the CDLP to a near optimal solution. The long running times reported in Table 5.3 are explained by the end of convergence that does not improve significantly the solution. For low quit ratio, the MBGP can slightly improves the revenue by adjusting customers last choices. For high quit ratio there is less room for maneuver because customers rapidly quit.

In Figure 5.7, we also report the MBGP iterative resolution for two building strategies $\widetilde{\text{MBGP}}_{\text{AR}}$ and $\widetilde{\text{MBGP}}_{\text{ARP}}$ as defined in Section 5.5.3.

The main observation is that iterative resolutions $\widetilde{\text{MBGP}}_{\text{AR}}$ and $\widetilde{\text{MBGP}}_{\text{ARP}}$ perform as well as the CDLP and better than the MBGP. It proves that refining the demand progressively allows us to supply a near optimal solution very quickly. It is faster than the CDLP branch and bound. First because hierarchy variables are effectively determinate in this iterative method. Secondly, because an important part of the buying graph has an insignificant impact on the problem.

5.6.3 Airline

This instance, illustrated in Figure 5.8, is based on the network of the Delta Air Lines limited to the eight major US airports in terms of annual traffic. There is a total of 306 flights each corresponding to a resource. These flights are sold through 8800 products. 1925 segments are interested to eventually buy a product. Each considers between 1 and 12 products with an average of 9.8. This instance is completely described in Section B.3 of the appendices.

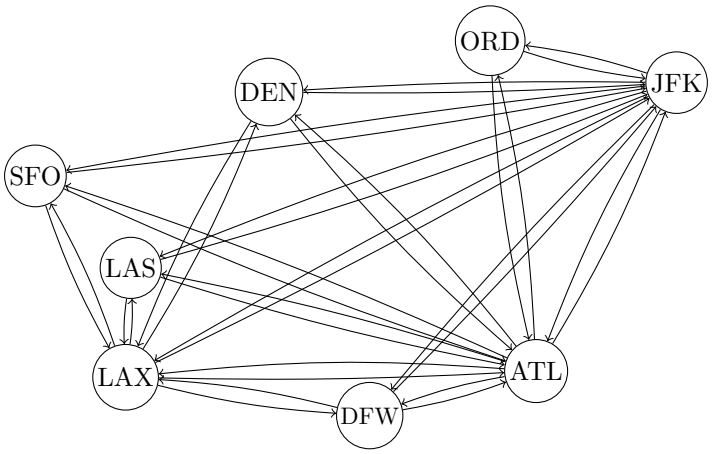

Figure 5.8 Markets of the Airline instance.

We do not benchmark the OD policy because it is even more intractable for this instance. The buying graph being too large to be built, we only use the $\widetilde{\text{MBGP}}$. To lighten the following figures, we only use the arrival ratio (AR) as strategy for the iterative resolution.

And because results are similar for the arrival ratio × price in the Bus-line instance.

Figure 5.9 summarize the CDLP column generation and $\widetilde{\text{MBGP}}_{\text{AR}}$ iterative resolution toward time for the first three hours of solving. Both convergences are obtained by cherry picking few points to ease the readability. At each point, we report the ideal revenue R, the expected revenue E[R] and three buying graph indicators: arrived, the revealed and potential percents, as seen in Section 5.5.3. MBGP_{AR} is the ideal revenue of the policy returned by the $\widetilde{\text{MBGP}}_{\text{AR}}$ but for the full demand. It is easily calculable by building the full buying graph knowing what is the policy.

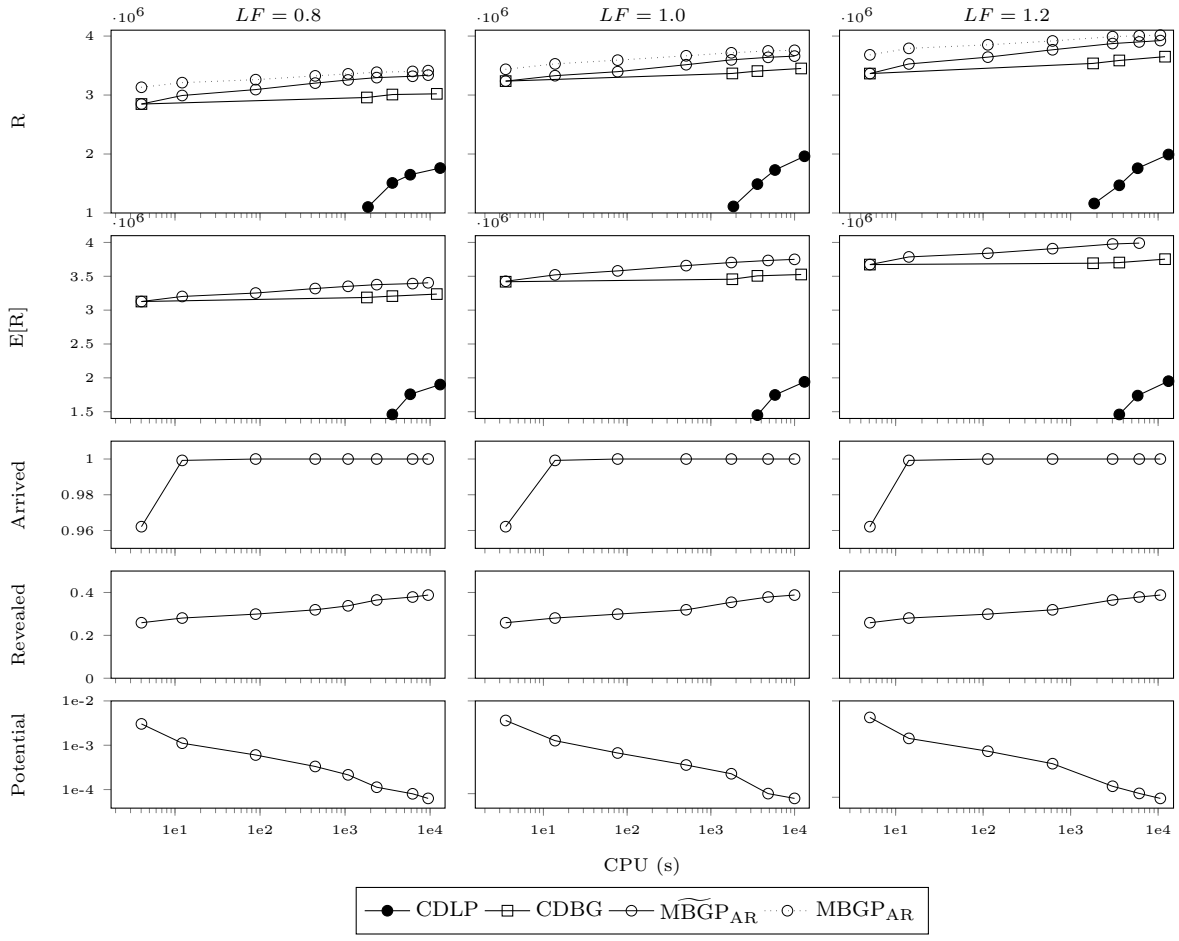

Figure 5.9 CDLP columns generation and MBGP iterative resolution by arrival ratio ($\widetilde{\text{MBGP}}_{\text{AR}}$) for the Airline instance with $w_{\emptyset}^{\%} = 5$. The time limit is 3 hours.

The main observation is that the $\widetilde{\text{MBGP}}_{\text{AR}}$ outperforms the CDLP for both the running time and the expected revenue. Our approach returns a solution in less than 5 min generating an expected revenue 2.5 times better than the CDLP solution obtained after 3 hours of column generations. In fact the $\widetilde{\text{MBGP}}_{\text{AR}}$ starts by focusing on the most important part of the demand and then refines the comportments to gain in robustness. On the other hand, the

CDLP solution returned after 3 hours is not usable. The poor performance of the CDLP is explained by the high number of columns to generate and the time to solve the subproblem. Whether for the greedy heuristic or the mixed exact formulation, the subproblem takes at least 2 min to be solved. Having 306 capacity constraints, we can assume without taking too much risk that the CDLP necessitates at least 306 column generations. It corresponds to $306 \times 3 \approx 10$ hours of solving.

Contrarily to previous instances, neither convergence reaches the optimal solution in the time allowed. Nothing prevents a higher solution than the one we observe. So that the CDLP could finally be faster in the long term. However, the CDLP solution after 50 hours of solving is only 1% better than the $\widetilde{\text{MBGP}}_{\text{AR}}$ reported here.

We note that only 33% of customers are completely revealed while 100% arrived. Even with that our method performs well because the remaining buying paths in the graph have a very low arrival ratio. In fact the potential indicator decreases by a factor 100. So that the last nodes added are not influencing a lot the solution because they are insignificant. Moreover, customers have surely their first, second or third choice satisfied. This potential indicator is thus very useful to judge when to stop the convergence of our approach. It explains why the difference between the MBGP_{AR} and $\widetilde{\text{MBGP}}_{\text{AR}}$ decreases as the iterative resolution advance.

Motivated by the good solution quickly returned by the $\widetilde{\text{MBGP}}_{\text{AR}}$ we also reported the CDBG. The starting point corresponds to the first point of the $\widetilde{\text{MBGP}}_{\text{AR}}$ iterative resolution. We note that our approach considerably accelerates the CDLP column generation and that $\widetilde{\text{MBGP}}_{\text{AR}}$ has a better convergence rate at that point.

5.7 Conclusion

We have presented a new approach for the CNRM problem. We focus on when to close each product sale, whereas most current approximations determine for how long to sell each offer. Our PC approach can be naturally ordered, avoiding some postprocessing.

We build our model by considering possible customer behavior. We thus obtain a buying tree, and we apply the PC times to it to determine for how long each product is sold so that we can formulate our nonlinear BGP. It is an approximation of the exact DP formulation of the problem. We show that the CDLP is an upper bound on the BGP, and that they are equal when the optimal CDLP policy has no reopening. In this case our BGP inherits results proved by Liu et van Ryzin (2008) about the DP upper bound and asymptotic optimality. The way the model is built allows us to consider any choice behavior. We can handle different types simultaneously without adaptation, even when the segments overlap.

We also show that the buying tree contains at most $\lfloor e\, n_L! \rfloor$ nodes, and the model obtained has at most $2^n - 1$ variables, which is the number of CDLP variables. Because the model exploits the segment structure, these numbers are often well below these upper bounds. However, the numbers can be significant for large instances, and the CDNP is solved in practice via an equivalent mixed integer linear program, the MBGP. We propose a method, denoted $\widetilde{\text{MBGP}}$, to solve the MBGP for a buying tree built progressively by adding nodes and increasing the number of customers the node represents.

Our experiments show that MBGP and CDLP return similar solutions. The expected revenue for MBGP is about 1% better when we simulate the PC policy because this policy is natural. For the largest instances, $\widetilde{\text{MBGP}}$ returns a good solution noticeably faster than CDLP can. By adding the customer behavior progressively, we develop a method that is more effective than the column generation of the CDLP for the instances tested. Our method also allows flexible choice behavior. It therefore seems appropriate for real-world networks for which the solution time is an important consideration.

Acknowledgment

The authors are grateful to the natural sciences and engineering research council of Canada (NSERC), the Fonds de recherche du Quebec en nature et technologies (FRQNT) and the company ExPretio technologies for having funded and supported this research. The authors also thank the anonymous referees and associate editor, whose feedback helped improve the paper.

CHAPITRE 6 ARTICLE 3: FLUID ARRIVALS SIMULATION FOR CHOICE NETWORK REVENUE MANAGEMENT

Le texte de ce chapitre est celui de l'article *Fluid arrivals simulation for choice network revenue management* accepté à *Journal of Revenue and Pricing Management.* Auteurs : Thibault Barbier, Miguel F. Anjos, Fabien Cirinei et Gilles Savard.

Abstract

Since the beginning of revenue management, simulation has been used to estimate the expected revenue resulting from an availability policy. It has also been used to verify the quality of forecasts by projecting them onto past availability policies. Recently, it has been used in simulation-based optimization approaches to find the best policy. Simulation thus has a central role in revenue management. We focus on the choice network revenue management (CNRM) problem that incorporates multiple resources and customer behavior. The traditional CNRM simulation is based on discrete customer arrivals; we propose a new approach based on fluid arrivals. Our estimator is biased, but we observe that the bias is often insignificant in practice and appears to be asymptotically null. Our approach consistently outperforms the traditional simulation in terms of estimation time and is thus a better choice for large instances. We also prove that it is equivalent to an approximation for the CNRM availability policy optimization problem. This equivalence limits the value of simulation-based optimization methods but allows us to propose heuristics to rapidly support the optimization.

6.1 Introduction and literature

Revenue management (RM) aims to match offers to demand, given limited and perishable resources, in order to maximize revenue. The resources are sold as products whose availability is controlled over the reservation period. Choice network revenue management (CNRM) considers resources and customer behavior simultaneously. The CNRM variant includes customer buying logic such as *buy-up* and *buy-down* and allows customers to *buy-across* resources. See Talluri et van Ryzin (2004b) for a complete review of RM. In this article, we focus on simulation for CNRM.

Simulation has been widely used in CNRM. First, it can measure the quality of the availability policies returned by optimization models. The CNRM availability policy optimization problem can be formulated as a dynamic program (DP; Liu et van Ryzin, 2008). The goal is

to manage the product availability over the reservation period in order to generate the highest revenue. However, the DP rapidly becomes intractable, and approximations are used; they must find a balance between simplicity and realism. Simulation can then be used to estimate the expected revenue resulting from an availability policy. This is an indicator of the performance of the approximation model. For example, Bront *et al.* (2009) benchmarks static and dynamic approximations based on the simulated expected revenue. However, simulation often requires many evaluations, even for small instances. Meissner *et al.* (2013) obtain a precision of approximately 6% of the expected revenue after 2000 evaluations for an instance with only six products. If greater accuracy is required, the simulation will be slow or intractable.

Second, simulation is used within some CNRM approximation models. Talluri (2010), Kunnumkal et Topaloglu (2011), and Talluri (2014) solve a randomized approximation several times, simulating the customer arrivals to tighten their solutions. The simulation adds a stochastic component to deterministic approximations.

Third, simulation is used to forecast demand and predict behavior. It can apply forecasts to historical availability policies to determine what products are booked. The resulting bookings are then compared to the actual bookings made to evaluate the forecast accuracy, and the forecast can be modified by integrating the insights from the simulation. Cleophas *et al.* (2009) develop a simulation framework with an artificial demand generator to compare the performance of forecasting methods. Fiig *et al.* (2014) propose a forecast accuracy measure for behavioral demand based on historical observations. They minimize the corresponding error to optimize the forecast parameters. In practice, simulation must be able to process large historical data sets in a reasonable time. Simulation is also used for the creation of training data with which to test methods, as in van Ryzin et Vulcano (2015) and van Ryzin et Vulcano (2017).

Fourth, simulation-based optimization methods have been explored for CNRM availability policy optimization because they accurately model the problem. Bertsimas et de Boer (2005a) and van Ryzin et Vulcano (2008b) develop stochastic gradient descent for RM without choice behavior. Van Ryzin et Vulcano (2008a) and Chaneton et Vulcano (2011) generalize the method to CNRM for nonparametric choice behavior. Other researchers propose model-free methods. For example, Bijvank *et al.* (2011) integrate a stochastic gradient technique while Gosavi *et al.* (2005) experiment with simulated annealing and simultaneous perturbation (SP) methods. simulation-based optimization approaches generally achieve only local convergence. Moreover, they require many evaluations of gradients or finite differences. For the largest instances, current simulation models are too slow.

Fifth, simulation can be used as a *what-if* tool to support decisions in CNRM. We can change one or more features (e.g., the availability policy or the resource capacity) and measure the changes in terms of revenue, bookings, load factor, or any relevant output. RM analysts use this to select promotions or group reservation deals. The Passenger Origin-Destination Simulator (PODS) was introduced by Boeing in the 1990s (Belobaba et Hopperstad, 1999) to analyze customers' RM preferences. It has since been further developed (Carrier, 2003; Weatherford, 2013; Carrier et Weatherford, 2015) and now belongs to PODS Research LLC. Eguchi et Belobaba (2004) use PODS to highlight the importance of group bookings for the Japanese airline market. Gorin et Belobaba (2004) use the software to investigate the potential of RM in a low-fare airline. Darot (2001) studies RM for airline alliances with PODS, and Frank *et al.* (2008) explain how to set up a stochastic simulation model for RM analyses. Frank *et al.* (2006) use simulation to measure the effects of continuous capacity adjustments for different allocation times. Doreswamy *et al.* (2015) use the Airline Planning and Operations Simulator (APOS) developed by Sabre to explore the impact of different RM methods. Bijvank *et al.* (2011) developed a complete Java simulation library for CNRM. When used as a what-if tool, simulation must quickly return an accurate expected revenue.

Simulation usually handles discrete customer arrivals; we refer to this as discrete arrivals simulation (DAS). The arrival process is stochastic, and the expected revenue is estimated by a Monte-Carlo approach (Gilks *et al.*, 1995). In this approach, each evaluation considers a random discrete arrival sequence. We calculate the revenue by applying the availability policy to the sequence. We then average the revenues obtained to estimate the expected revenue. This estimator is unbiased and approaches the real expected revenue as the number of evaluations increases. The precision relies on the confidence interval (CI), which is proportional to the ratio between the evaluation variance and the root square of the number of evaluations. The variance depends on many complex factors and is thus difficult to calculate a priori. Increasing the number of evaluations will improve the accuracy of the revenue estimate. However, each evaluation must process every arrival in the DAS model, so this estimator is slow for large instances.

In this article, we introduce another way to use simulation to estimate the expected revenue in CNRM: we consider a continuous flow of arrivals. Our approach is called fluid arrivals simulation (FAS). It has been used by Kesidis *et al.* (1996) for ATM networks and by Figueiredo *et al.* (2006) for computer networks. To the best of our knowledge it has not been applied to CNRM.

FAS does not use the Monte-Carlo technique because it estimates the expected revenue in a single evaluation by neglecting the order of the arrivals. It takes about the same time as a

few DAS evaluations. Consequently, FAS outperforms DAS in terms of estimation time in all our experiments. Because the estimation is direct, FAS is also invariant, but it is biased. This bias is difficult to determine in theory because of the mechanism of the behavior and availability policy. In practice, it is relatively small for large instances and seems to be asymptotically null.

We prove that FAS is equivalent to the choice deterministic linear program (CDLP; Liu et van Ryzin, 2008), which is a widely used approximation for CNRM availability policy optimization. This equivalence limits the value of simulation-based optimization methods for FAS because it is preferable to directly solve the CDLP. We conduct experiments that show the slow convergence of an SP method for FAS and the need for a good initial solution. However, this equivalence allows us to develop new approaches to support the solution of the CNRM problem. They benefit from the speed of FAS and help to reduce the solution time. The two approaches that we propose are: the selection of a good initial CDLP solution by simulation and the estimation of the CDLP policy for a simplified demand. Both methods are simple and greatly accelerate the CDLP in our experiments. Our estimator therefore potentially has a wide range of applications.

The remainder of this paper is organized as follows. In Section 6.2, we present the CNRM notation and discuss discrete arrivals. We then present DAS, which is the traditional estimator of expected revenue in CNRM. Section 6.3 presents our FAS estimator. We describe the discrete changes that occur although the simulation is considered fluid. We then analyze the properties of the bias. In Section 6.4 we examine the use of our estimator for the CNRM availability policy problem. We start by presenting an SP algorithm for simulation-based optimization with FAS. We then prove that FAS is equivalent to CDLP, and finally we suggest some ways to support optimization with our estimator. Our computational experiments are reported in Section 6.5, and Section 6.6 provides concluding remarks.

6.2 Simulation for the CNRM

In this section, we start by giving the principal CNRM definitions. We then describe the process by which individual customers arrive during the reservation period and eventually buy a product. We finish by presenting the DAS estimator, which is the traditional simulation for CNRM.

6.2.1 Definitions

CNRM is based on resources $i \in I$, each with a capacity c_i. There are $m = |I|$ resources. These resources are incorporated into products $j \in J$ that are sold at a fare r_j. There are $n = |J|$ products. We denote by I_j the set of resources consumed by each product j. We denote by $S \subseteq J$ a set of products, and we call it an offer. These products are sold during the reservation period, from time $t = 0$ to time $t = T$. The resources perish at the end of the reservation period ($t = T$).

The goal is to control the availability of the products over the reservation period in order to maximize the revenue generated by the sales. Let $x \leq c$ be the vector of remaining capacities and $J(x) \subseteq J$ the set of products with nondepleted resources. We must find the availability policy formed by offers $O(t, x) \subseteq J$ for all $t \leq T$ and $x \leq c$ that maximizes the revenue. The set of products available at any time and for any remaining capacity is $S(t, x) = O(t, x) \cap J(x)$.

A segment $l \in L$ groups customers with the same choice behavior who are interested in the same products $C_l \subseteq J$. This behavior is reflected by the probability $P_l(j|S)$ of buying product j if offer S is proposed. The customers of a segment arrive during the reservation period according to a Poisson process with arrival rate λ_l (a constant). We define $\lambda(S) = \{\lambda_j(S)\}_{j \in J}$ to be the product arrival rate vector if offer S is proposed, where $\lambda_j(S) = \sum_{l \in L} \lambda_l P_l(j|S)$ for each component.

We use the *running instance* (RI) of Figure 6.1 to explain the following concepts and models. The reservation period is $T = 1$, and the availability policy offers the first product for 0.3

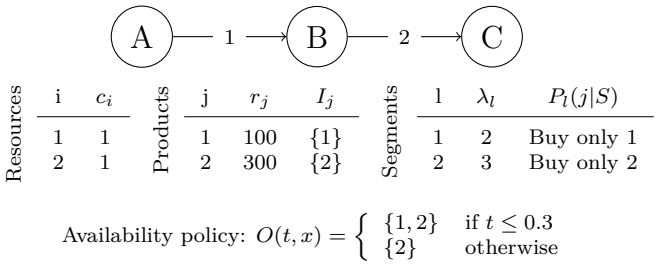

Figure 6.1 Running instance.

periods and the second throughout the reservation period.

6.2.2 Arrivals process

The customers arrive over the reservation period according to a random Poisson process by segment. The segment arrival sequence is therefore a random process; we denote by Ω its

set of realizations. Let $\omega \in \Omega$ be a random segment arrival sequence with $\overline{\omega}$ distinct arrivals. It is indexed by $k \in [1, \overline{\omega}]$ to identify each arrival chronologically. The kth arrival occurs at $t_\omega^k \in [0, T]$ for remaining capacities $x_\omega^k \leq c$ and corresponds to a customer of segment $l_\omega^k \in L$. We have $t_\omega^k \leq t_\omega^h$ and $x_\omega^k \geq x_\omega^h$ for all $h \in [k, \overline{\omega}]$.

Let $\psi(l, S)$ be the random purchase function, returning a purchase vector $\{\psi_j(l, S)\}_{j \in J}$. The sole component equal to one corresponds to the product bought. The set of realizations is Ψ, and the purchase function satisfies

$$E_\Psi \left[\psi(l, S)\right] = \{P_l(j|S)\}_{j \in J}. \tag{6.1}$$

We can calculate the random revenue R_ω^k generated by the kth arrival of any arrival sequence ω as follows:

$$R_\omega^k = R_\omega^{k-1} + r^\top \psi \left(l_\omega^k,\, S(t_\omega^k, x_\omega^k)\right), \quad \forall k \in [1, \overline{\omega}], \omega \in \Omega$$

where $R_\omega^0 = 0$. The random revenue R_ω of the entire arrival sequence ω is then:

$$R_\omega = \sum_{k=1}^{\overline{w}} r^\top \psi \left(l_\omega^k,\, S(t_\omega^k, x_\omega^k)\right), \quad \forall \omega \in \Omega. \tag{6.2}$$

We determine the CNRM expected revenue as follows:

$$E[R] = E_{\Omega \times \Psi}[R_\omega]. \tag{$E[R]$}$$

As mentioned in Section 6.1, this expected revenue is fundamental for CNRM because it reflects the revenue received in practice. However, the combination of two realization sets, the availability policy, and the overlapping segments makes it almost impossible to calculate the expected revenue.

The calculation is possible for our small RI. Segment 1 arrives at least once between 0 and 0.3 with probability $1 - e^{-2\times 0.3}$, and segment 2 arrives at least once over the reservation period with probability $1 - e^{-3}$. The expected revenue is therefore $(1 - e^{-2\times 0.3}) \times 100 + (1 - e^{-3}) \times 300 = 330.18$.

6.2.3 Discrete arrivals simulation (DAS)

The expected revenue is usually estimated, and the traditional RM approach is based on the Monte-Carlo method. Instead of a complete enumeration, this method draws N segment arrival sequences ω^k with $k \in [1, N]$. A revenue $\widetilde{R}_{\omega^k}$ for each sequence is obtained from Eq. (6.2) by choosing a random purchase. The expected revenue is then obtained by averaging

these revenues:

$$\mu^D = \frac{1}{N}\sum_{k=1}^{N} \widetilde{R}_{\omega^k} \xrightarrow[N\to\infty]{} E[R]. \tag{DAS}$$

This DAS estimator computes each discrete customer arrival. According to the strong law of large numbers, μ^D converges almost surely to $E[R]$ as the number of arrival sequence increases. This estimator is therefore unbiased.

The central limit theorem gives an α CI CI_α^D for this estimator:

$$\mathrm{CI}_\alpha^D = \frac{\delta_\alpha\sqrt{\mathrm{Var}[R]}}{\sqrt{N}}, \quad \forall\alpha \in [0,1] \tag{CI}$$

where γ_α is the $\frac{1-\alpha}{2}$ percentile of the normal distribution and $\mathrm{Var}[R]$ is the variance, which measures the volatility of the revenue. The CI establishes that $\alpha\%$ of the values are in $\left[\mu^D - \frac{\mathrm{CI}_\alpha^D}{2}, \mu^D + \frac{\mathrm{CI}_\alpha^D}{2}\right]$. The precision of the DAS is thus inversely proportional to the square of the number of evaluations.

The variance is almost impossible to calculate and is thus estimated by the sample variance:

$$\sigma^D = \frac{1}{N}\sum_{k=1}^{N}\left(\widetilde{R}_{\omega^k} - \mu^D\right)^2 \xrightarrow[N\to\infty]{} Var[R].$$

The speed of the DAS convergence depends on this variance. If the variance is high, many evaluations are necessary to increase the precision, as we can see in Eq. (CI).

Each arrival requires the central reservation system (CRS) to process the available products. The complexity of each evaluation is therefore proportional to the average number of CRS calls per evaluation, denoted CRS^D. Each evaluation covers a sequence of $\overline{\omega}$ arrivals, so $\mathrm{CRS}^D \approx E_\Omega[\overline{\omega}]$. The overall DAS complexity therefore depends on the revenue variance and the number of CRS calls per evaluation. Unfortunately, the arrival stochasticity and the availability policy logic often lead to a high variance. Moreover, for large instances the number of arrivals, and thus of CRS calls, can be considerable. DAS may be unable to estimate the expected revenue accurately in the time allowed.

We plot in Figure 6.2 the DAS estimation convergence of the RI expected revenue with a 95% CI. We observe that DAS is unbiased, as expected.

6.3 Fluid arrivals simulation (FAS)

In this section, we present our new approach to estimate the expected revenue. It is inspired by work on fluid simulation for queues and computer networks. We also detail here its

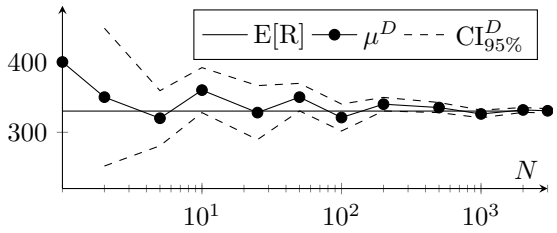

Figure 6.2 DAS convergence for the RI.

mechanisms and properties.

6.3.1 Model formulation

For DAS, the order of the arrivals leads to stochasticity. The main idea of our approach is to consider the arrivals of each segment as a fluid rather than individuals. For example, five customers of a segment arriving over a reservation period of two intervals are considered as a segment arriving with a rate of 5/2. The order of the arrivals thus becomes unimportant, and the expected revenue can be calculated as follows:

$$\mu^F = E_\Psi \left[\int_{t=0}^{T} \sum_{l \in L} \lambda_l \, r^\top \psi \left(l, S(t,x) \right) \delta t \right] = \int_{t=0}^{T} \sum_{l \in L} \lambda_l \; r^\top E_\Psi \left[\psi \left(l, S(t,x) \right) \right] \delta t.$$

In this continuous case, the expected value of ψ is simplified as in Eq. (6.1) and the FAS is

$$\mu^F = \int_{t=0}^{T} r^\top \lambda(S(t,x)) \delta t. \tag{FAS}$$

The FAS estimator is therefore continuous and deterministic because it is calculated in a single evaluation ($N^F = 1$). It is also invariant ($\sigma^F = 0$).

6.3.2 Discrete changes

FAS is continuous but the function $S(t,x)$ is a set of products with discrete additions and removals. We assume that the number of changes is finite, which is the case in practice. We index by $k \in K$ these changes over time where K is the set of changes of size $n_K = |K| - 1$. Each change occurs at time t_k and for the remaining capacities x_k. The first change, $k = 0$, corresponds to the start of the reservation period: $t_0 = 0$ and $x_0 = c$. The final change, $k = n_K$, corresponds to the end of the reservation period: $t_{n_K} = T$ and $x_{n_k} = 0$. Between two changes, the set $S(t,x)$ is constant and denoted by $S_k = S_k(t_k, x_k)$. We can now rewrite

the FAS calculation as follows:

$$\mu^F = \sum_{k=0}^{n_K} \int_{t=t_k}^{t_{k+1}} r^\top \lambda(S_k)\, \delta t = \sum_{k=0}^{n_K} r^\top \lambda(S_k)\, d_k \tag{6.3}$$

where $d_k = t_{k+1} - t_k$ is the time between two consecutive changes. There are three possible changes:

— A resource depletion that changes $J(x)$;

— A change in availability policy $O(x, t)$;

— The end of the reservation period, $S(t, x) = \emptyset$.

When a change occurs, we can easily determine the next change by calculating the minimum time step to the next resource depletion, the next change in the availability policy, or the end of the reservation period.

For the RI, the first change corresponds to the beginning of the reservation period. The offer is $S_0 = \{1, 2\}$ and the product arrival rate is $\lambda_0 = (2\ 3)$. We then calculate the above time steps; they are indicated by an $\times$ in Figure 6.3. The minimum time step is $d_0 = 0.3$,

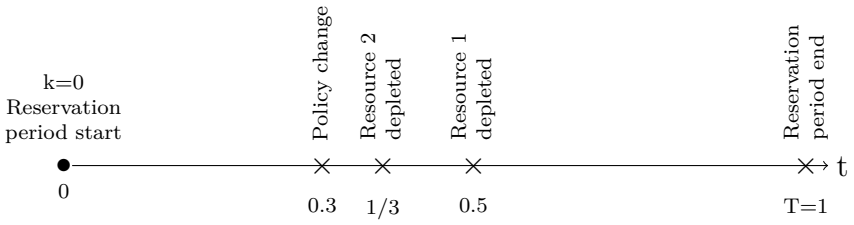

Figure 6.3 Determination of second change for RI.

corresponding to the policy change at $t = 0.3$. We have sold $d_0\lambda_0$ products between these two changes, and we have $t_1 = 0.3$, $S_1 = \{2\}$ and $\lambda_1 = (0\ 3)$. We illustrate the possibilities for the third change in Figure 6.4. The minimum time step corresponds to the depletion of

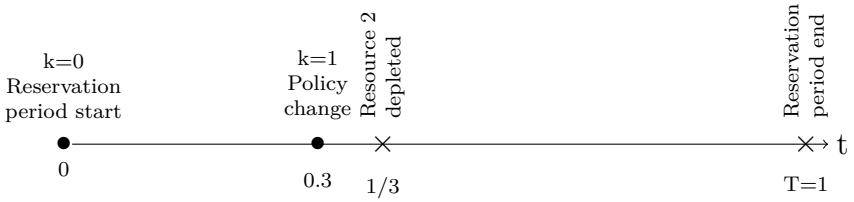

Figure 6.4 Determination of third change for RI.

resource 2 at $t = 1/3$. The final change is the end of the reservation period. We have sold $d_0\lambda_0 + d_1\lambda_1 + d_2\lambda_2 = 0.3(3\ 2) + (1/3 - 0.3)(0\ 3) + (1 - 1/3)(0\ 0) = (0.6\ 1)$ products and $\mu^F = 360$.

The complexity of the FAS depends on the number of changes. At each change the CRS must compute the minimum time step to the next change. This CRS call, denoted CRS^F, may be more complex than that for DAS. It is almost impossible to determine CRS^F a priori, but in some cases we can find a bound on the number of changes. If there is no reopening of product sales over the reservation period, the only possible changes are resource depletion, product removal, and the end of the reservation period. Each of these changes occurs only once, so $CRS^F \leq n + m + 1$.

6.3.3 Bias and properties

With the RI, the estimate $\mu^F = 360$ is not equal to the theoretical expected revenue, $E[R] = 330.18$. This shows that FAS is a biased estimator of the CNRM expected revenue. We denote the FAS bias by $\theta^F = E[R] - \mu^F$ and approximate it by $\widetilde{\theta}^F$ as follows:

$$\widetilde{\theta}^F = \mu^D - \mu^F \xrightarrow[N\to\infty]{} \theta^F. \qquad \text{(FAS Bias)}$$

The relative estimated bias is $\Delta\widetilde{\theta}^F = \frac{\mu^D - \mu^F}{\mu^D}$. The bias is explained by the situations where a discrete resource capacity is sold in fractional quantities to multiple customers. In contrast, in the DAS model a resource cannot be partially sold. For the FAS of the RI, the first resource is sold to 0.6 customers of the first segment.

The number of fractional situations depends on the instance. It is difficult to predict because it depends on the number of resources, the policy, and the demand.

We now show that FAS can underestimate ($\theta^F \geq 0$) as well as overestimate ($\theta^F \leq 0$).

Proposition 10. *The FAS bias can be positive or negative.*

Proof of Proposition 10. Consider two resources with $c_1 = 2$ and $c_2 = 1$, and two products with prices r_1 and r_2 such that $I_1 = \{1\}$ and $I_2 = \{1, 2\}$. Suppose there are two segments, both arriving at the rate 2 during a reservation period T =1. We can easily show that $E[R] \approx \frac{7}{6}r_1 + \frac{5}{6}r_2$ and $\mu^F = r_1 + r_2$. Hence, $\theta^F = \frac{7}{6}(r_1 - r_2)$. By adjusting the values of r_1 and r_2, we obtain either positive or negative bias. □

Proposition 11. *The FAS bias can be arbitrarily large.*

Proof of Proposition 11. By adjusting r_1 and r_2 in the proof of Proposition 10, we obtain an arbitrarily large bias. □

However, in practice the size of the bias is reasonable. Moreover, the relative bias is $\Delta\widetilde{\theta}^F = \frac{\theta^F}{\mu^D} = \frac{r_2 - r_1}{7r_1 + 6r_2}$, which is in the range $-\frac{1}{7} \leq \Delta\widetilde{\theta}^F \leq \frac{1}{6}$ and is thus relatively insignificant. It is difficult to theoretically determine the bias because it depends on the policy, the arrival stochasticity, and the resource capacity. Furthermore, it is mainly caused by phenomena occurring when one or more resources have a capacity close to one.

6.4 FAS and optimization

In this section, we focus on how FAS can solve or support the CNRM availability policy problem. We start by presenting an SP algorithm for FAS. We then prove the equivalence between FAS and one of the principal CNRM approximations. We finally propose two simple methods that use FAS to support the solution of this problem.

6.4.1 Simulation-based optimization

One of the most widely used methods in simulation-based optimization is the SP algorithm; see Spall (1998) for more details. For this algorithm, we use a product closing (PC) availability policy. It specifies a time $0 \leq t_j \leq T$ to close the sale of each product such that:

$$S^{PC}(t, x) = \{j \mid j \in J, t_j \geq t\}.$$

SP is a gradient descent method. We denote by t^k the vector of products closing at iteration k. This technique is based on the following approximation:

$$\frac{\partial \mu(t)}{\partial t_j}|_{t=t_k} \approx \frac{\mu(t + h) - \mu(t - h)}{2h_j}$$

where $h = \{h_j\}_{j \in J}$ and $h_j = \frac{B}{k^\beta}$ with B a Bernoulli random variable and $\beta \in [0, 1]$ a tuning parameter. The next PC policy is thus obtained as follows:

$$t_j^{k+1} = \Pi_{0 \leq t \leq T}\left[t_j^k + \alpha^k \frac{\partial \mu(t)}{\partial t_j}|_{t=t_k}\right], \quad \forall j \in J$$

where $\alpha^k = \frac{\alpha}{k}$ is the step size of the descent.

We did not study the properties of the function $\mu(t)$; see Spall (1998) or Gosavi (2015) for the convergence properties of SP.

6.4.2 Equivalence to CNRM optimization

Static approximations of the CNRM availability policy optimization problem avoid the discrete customer arrival complexity of the DP by considering a continuous and deterministic flow of customers. They all have the same structure:

$$\begin{aligned} R = \quad & \max_{q} r^\top q \\ s.t. \quad & Aq \leq c, \\ & q \geq 0. \end{aligned} \tag{STATIC}$$

The CDLP is a static approximation based on an availability policy that controls the time $d_S \geq 0$ for which each offer must be proposed:

$$\begin{aligned} q = \quad & \sum_{S \subseteq J} \lambda(S) d_S \\ s.t. \quad & \sum_{S \subseteq J} d_S \leq T, \\ & d_S \geq 0, \quad \forall S \subseteq J. \end{aligned} \tag{CDLP}$$

The relationship between the CDLP and the DP is quite similar to the relationship between FAS and the exact E[R]. In both cases, the individual arrivals are replaced by their expected arrival rate. We can prove that the CDLP revenue is equivalent to the FAS estimation for any offer duration.

Proposition 12. $\mu^D = R^{CDLP}$.

Proof. The CDLP set of products with non-null duration $\widetilde{S}_u$ is arbitrarily ordered over $u \in [0, m-1]$ because the CDLP has at most m sets with non-null durations. Each set has a duration $\widetilde{d}_u$, giving $\widetilde{t}_{u+1} = \widetilde{t}_u + \widetilde{d}_u$ with $\widetilde{t}_{m+1} = T$ and $\widetilde{S}_{m+1} = \emptyset$. We apply Eq. (6.3) to this policy. The first change occurs at $\widetilde{t}_0 = 0$ and corresponds to the initial CDLP set $\widetilde{S}_0$. The next change does not occur until $\widetilde{t}_1$ because the STATIC capacity constraint ensures that no resources are depleted during this period, and the end of the reservation period is not reached because of the second CDLP constraint. We therefore prove by recurrence that we have $k = u$ until $k = m$ and hence

$$\mu^F = \sum_{k=0}^{n_K} r^\top \lambda(S_k)\, d_k = r^\top \left(\sum_{k=0}^{m} \lambda(\widetilde{S}_k)\, \widetilde{d}_k + \sum_{k=m+1}^{n_K} \lambda(\widetilde{S}_k)\, \widetilde{d}_k \right) = r^T \sum_{k=0}^{m} \lambda(\widetilde{S}_k)\, \widetilde{d}_k = R^{\text{CDLP}}.$$

For $k > m$, we necessarily have $\widetilde{S}_k = \emptyset$ to ensure the STATIC capacity constraint, so

$\lambda(\widetilde{S}_k) = 0.$ □

This proof was developed for the CDLP, but it is similar for many static approximations, so the equivalence is easily extended. The equivalence and the fact that the simulator is based on fluid arrivals show that FAS cannot improve the robustness of the static approximation solution by taking into account the arrival order stochasticity; this is in contrast to DAS.

6.4.3 Optimization support

Our approach can support the optimization although it does not improve the solution quality because of the arrival order stochasticity. Its rapidity and the equivalence result allow several applications. We propose two simple ideas:

— We can solve the CDLP for only the most significant part of the demand to reduce the solution time and then simulate the corresponding optimal policy by FAS to evaluate it for the full demand.

— We can use FAS with a metaheuristic to provide a good initial solution for the CDLP.

Both approaches are tested in the experiments.

6.5 Computational results

In this section, we perform numerical experiments on the following two estimators:

DAS is the traditional estimator described in Section 6.2.3. Recall that μ^D is the estimate of the expected value E[R], and σ^D is the estimate of the variance VAR[R]. CPU^D is the running time, and the number of evaluations is N^D. CRS^D is the number of CRS calls, which for this estimator is equal to the number of arrivals. To stop the convergence, we use a 95% relative CI width $\Delta[IC^D_{95\%}] = \frac{[IC^D_{95\%}]}{\mu^D}$ or a maximum number of evaluations.

FAS is our new estimator introduced in Section 6.3. Recall that μ^F is the estimate of the expected value E[R]. There is only one evaluation and thus no variance ($N^F = 1$, $\sigma^F = 0$). The running time is denoted CPU^F. CRS^F is the number of CRS calls, corresponding for this estimator to the number of changes.

Bus-line is an instance of eight buses leaving every two hours from 07:00 to 21:00 from city A to cities B, C, D, E, and F. A total of 15 markets are served, as illustrated in Figure 6.5. Each bus has a capacity of 30 and there are $5 \times 8 = 40$ resources. Two fares (low, high) are offered for each trip, giving a total of $15 \times 8 \times 2 = 240$ products. In the bus industry, tickets are usually available at least two months in advance, so we set $T = 60$

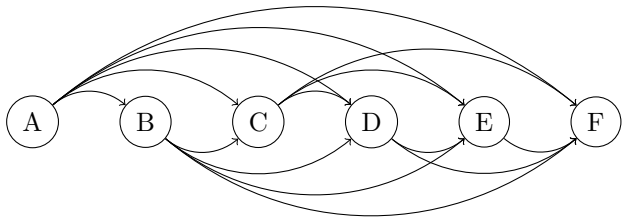

Figure 6.5 Markets for bus-line instance.

days. In total there are $15 \times 8 = 120$ segments with nonparametric choice behavior and on average 5.3 products.

Airline is an instance based on the Delta Air Lines network limited to the five major US airports. The network has 20 markets, as illustrated in Figure 6.6. It has 115 resources that correspond to the flights. There are 1591 products, and the reservation period is $T = 360$ days. There are 438 segments with nonparametric choice behavior and on average 7.9 products.

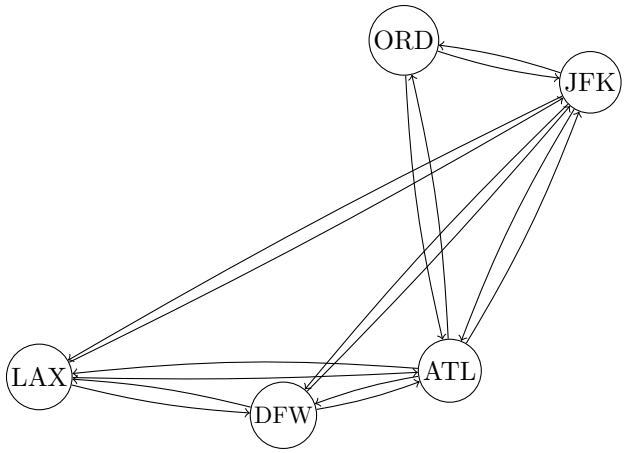

Figure 6.6 Markets for airline instance.

The load factor (LF) is defined as $\mathrm{LF} = \sum_{l \in L} \lambda_l / \sum_{i \in I} c_i$. We use PC times for the availability policy; these set the times when the sales of each product are closed.

6.5.1 Convergence and bias

In this section, we compare the convergence of the two estimators and analyze the bias of FAS. The term convergence is imprecise for FAS since it calculates the expected revenue in a single evaluation. However, this is a way to illustrate the differences between these two estimators. Figure 6.7 illustrates the convergence of the two estimators for the optimal

availability policy returned by the CDLP. This approximation is explained in Section 6.4.2. We stop the DAS simulation after 1000 evaluations.

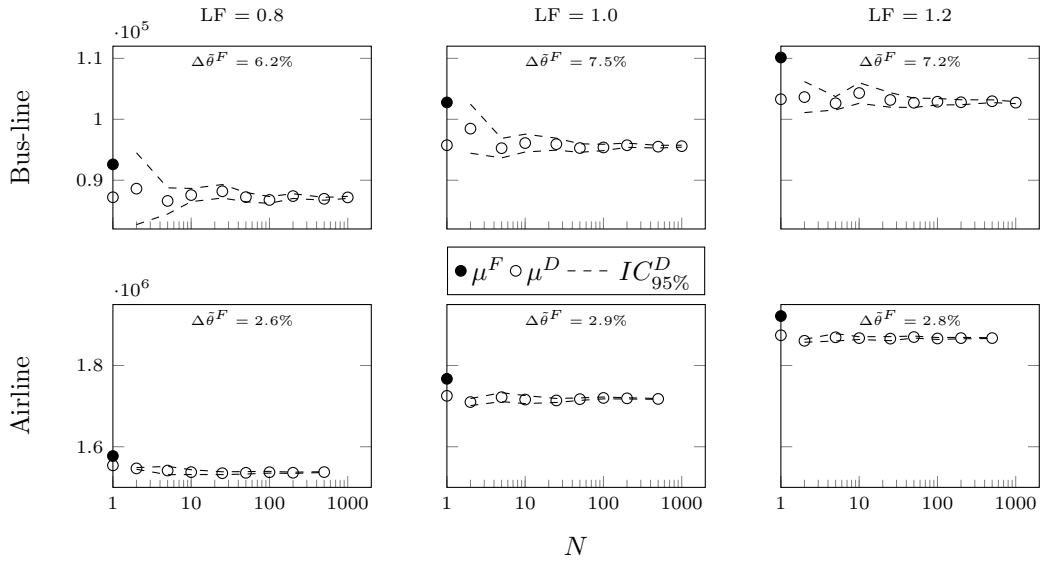

Figure 6.7 Expected revenue estimates μ^F and μ^D with respect to the number of evaluations N for the optimal CDLP availability policy. The FAS relative bias $\Delta\tilde{\theta}^F$ and the DAS 95% confidence interval $IC^D_{95\%}$ are also reported.

The most relevant observation is that FAS always overestimates the real expected revenue, by 6.9% on average for the bus-line and 2.8% for the airline. This overestimation arises because the availability policy is optimal. At optimality, the fluid aspect is emphasized because the optimizer takes it into account to maximize the revenue. The difference with DAS is thus at its peak because FAS returns exactly the optimizer revenue, as proved in Section 6.4.2.

We observe that the bias increases as the LF increases from 0.8 to 1.0 and is then approximately constant. This is verified by further experiments for the bus-line. With the notation (LF, $\Delta\widetilde{\theta}^F$), we report (0.2, 3.63%), (0.4, 4.24%), (0.6, 4.55%), (1.4, 7.39%), (1.6, 7.63%), and (1.8, 7.41%). Therefore, the bias becomes constant once a certain LF is reached. This could be because the optimizer does not improve the revenue with an additional fluid aspect. Another explanation is that the fluid aspect situations seen in Section 6.3.3 are all captured from a certain LF.

We also note that there is a difference of magnitude in the bias for the two instances. The bus-line bias is around 2.5 times higher than that for the airline. This suggests the asymptotic nullity of the bias: the average capacity per resource is 30 for the bus-line and 180 for the airline. The fluid aspect situations are more absorbed in the airline, clearly because of the higher capacity. This also highlights the difficult of predicting the bias.

To further investigate these poor results of FAS, we generate random availability policies for different scenarios. These scenarios vary the percentage γ_{close} of PC times fixed to zero; the other PC times are randomly chosen according to a uniform law. When $\gamma_{close} = 0$, every product is offered, and when $\gamma_{close} = 1$ no product is offered. We report in Figure 6.8 the relative FAS bias for different relative widths of the DAS CI and with respect to the scenarios for the PC times and the LF. We selected 100 and 25 availability policies per scenario respectively for the bus-line and the airline. The full results are reported in the Appendix: see Table C.2 for the bus-line and Table C.3 for the airline.

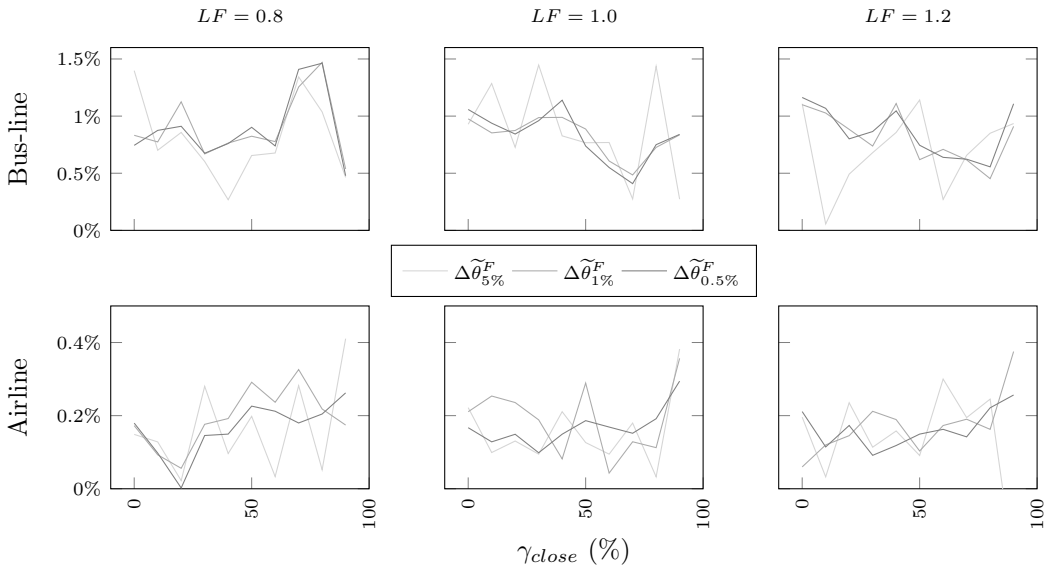

Figure 6.8 Relative bias $\Delta\widetilde{\theta}^F$ with respect to the percentage γ_{close} of closed products.

We first observe that the relative difference in the bias is not as high as before. It is 7.7 times lower (6.9% to 0.9%) and 14 times lower (2.8% to 0.2%) respectively for the bus-line and airline instances. This confirms that the optimal policy emphasizes the fluid aspect to maximize the revenue. Therefore, the optimal availability policy is certainly the one with the highest bias.

We observe that the bias evolution does not seem to follow any specific trend and might be unrelated to γ_{close}. This confirms that the fluid aspect does not depend on any one factor but is a consequence of a more complex interaction between the demand, the policy, and the structure of the instance.

We note that the bias for the 5% relative CI has considerable variability, whereas the 1% and 0.5% biases are smoother and similar. This shows that the DAS convergence is not rapid. For precise estimation, a 1% relative CI seems appropriate.

We now investigate the relationship between the FAS bias and the instance structure. We

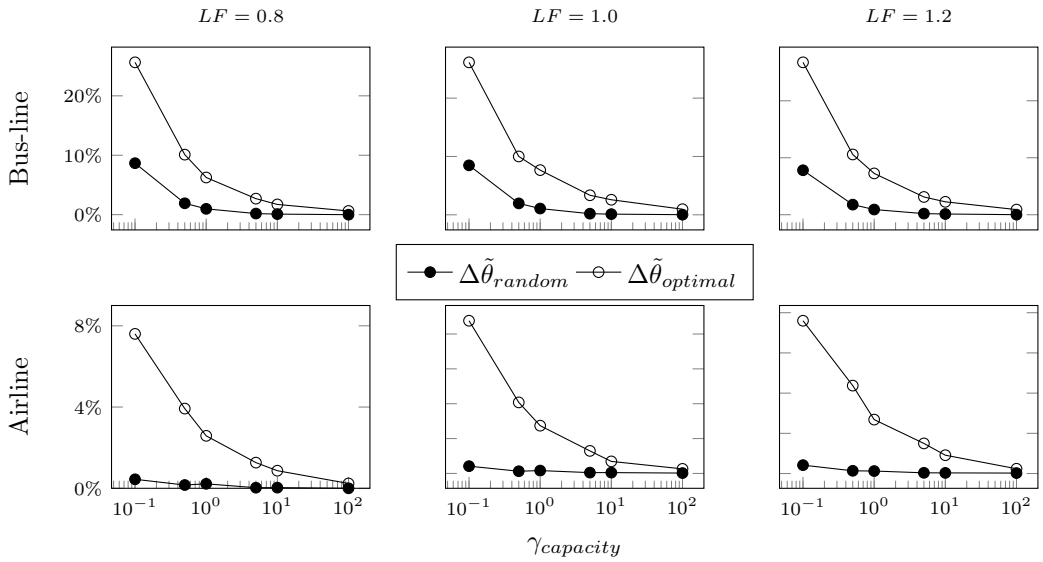

Figure 6.9 Relative bias $\Delta\widetilde{\theta}^F$ for the optimal and random policies when capacity is scaled and demand adjusted proportionally.

vary the capacity of both instances by a factor $\gamma_{capacity}$, i.e., $c \to c \times \gamma_{capacity}$. For each capacity scenario, we maintain the LF by scaling up the segment arrival ratio. In Figure 6.9, we report the evolution of the FAS bias for averaged random availability policies and for the optimal CDLP availability policy. The LFs are 0.8, 1.0, and 1.2. The capacity factor varies from 0.1 to 100. We first observe the same features as in the previous experiments. The optimal policy is always the one with the highest bias. It is on average between 3 and 30 times higher than the random policy bias. This figure also shows that the fluid aspect is more prominent in the bus-line instance (0 to 20%) than in the airline instance (0 to 8%). The main observation is that the bias decreases as the capacity and demand are scaled up. We have not proved that the bias is asymptotically null, but the results suggest this. However, an asymptotic result does not in practice determine the bias of an instance.

In conclusion, these experiments show that the bias is caused by a fluid aspect that is difficult to predict. It depends on a combination of mechanisms between the instance structure, the demand, and the policy. It is stronger when the availability policy is optimized. However, the bias is low for random policies, is reduced for instances with a higher capacity, and appears to be asymptotically null.

6.5.2 Estimation time

We now compare the estimators in terms of estimation time. Estimations must be returned quickly to allow users to rapidly test several options before making a decision. Also, these

estimators are often used in simulation-based optimization methods that require many estimations.

In Figure 6.10, we report the estimation times with respect to γ_{close} for the experiments of Section 6.5.1. The estimators compared are FAS and DAS for three relative CI widths: 5%, 1%, and 0.5%.

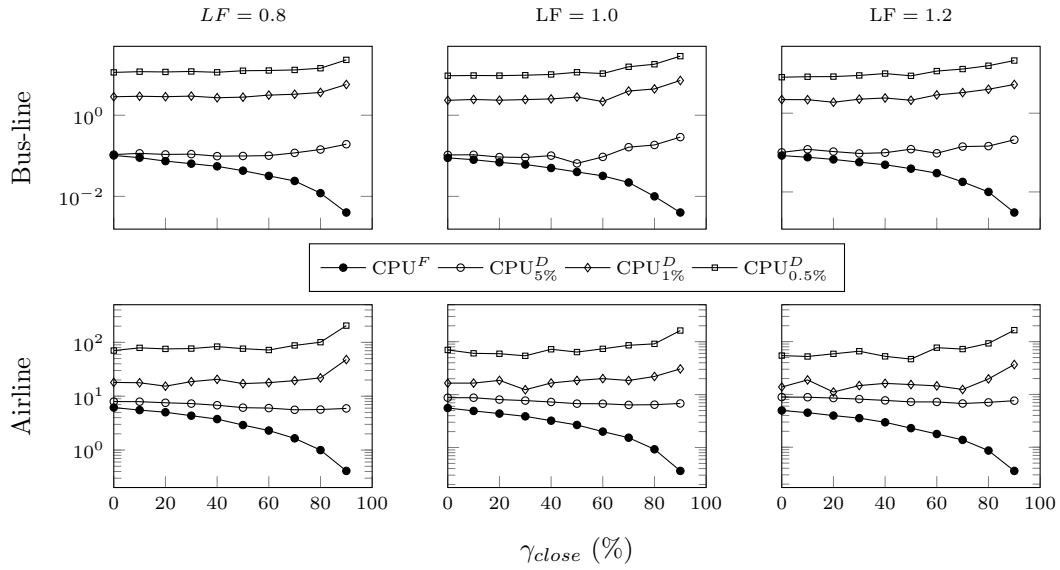

Figure 6.10 Estimation time for FAS and DAS ($[\text{IC}^D_{95\%}]$ is 5%, 1%, and 0.5%) with respect to γ_{close}.

The most important observation is that FAS is always faster, whatever the value of γ_{close} and the relative CI width. As expected, the difference increases with the relative CI width because DAS needs more evaluations to reach the necessary precision, as explained by Eq. (CI). Overall, FAS is faster than the 5%, 1%, and 0.5% DAS: respectively 2.7, 67.9, and 275.6 times faster for the bus-line and 2.4, 6.5, and 27.5 for the airline.

We note that the superiority of FAS is even more pronounced as γ_{close} increases. It is respectively 33.7, 87.8, and 1673.1 times faster for the bus-line and 3.4, 8.6, and 101.7 times faster for the airline when γ_{close} is 20%, 60%, or 90% in comparison with the 1% DAS. As γ_{close} increases, the number of changes decreases because more products are closed, and the relative DAS variance increases because the revenue depends on what products were closed. This is shown by the measures of variance and the CRS calls reported in Tables C.2 and C.3.

We observe that the estimation times are slightly lower for both estimators as the LF increases. For FAS and the LFs 0.8, 1.0, and 1.2, the average estimation times are respectively 0.05, 0.045, and 0.043 for the bus-line and 3.3, 2.9, and 2.6 for the airline. This is due to the higher demand that tends to consume resources faster and thus decreases the number of

changes, as shown in Tables C.2 and C.3. For the 1% DAS and the LFs 0.8, 1.0, and 1.2, the average estimation times are respectively 3.3, 3.2, and 2.9 for the bus-line and 92.6, 79.2, and 73.7 for the airline.

In Table C.1, we report the measures of the FAS and DAS estimations for the optimal availability policy. We note that FAS is 18.5, 34.1, and 53.3 times faster for the bus-line and 5.6, 9.6, and 6.3 times faster for the airline than the 1% DAS for the 0.8, 1.0, and 1.2 LFs. This supports our observation on the relationship between the estimation time and the LF.

As expected, the estimation of the bus-line expected revenue is 7.0 times faster than that for the airline. This is mainly because the airline instance is larger in terms of resources, products, and segments.

In Figure 6.11, we report the time to compute a CRS call for the two estimators with respect to γ_{close}. For FAS, each CRS call corresponds to a change, and we must compute the next change by calculating the time step. For DAS, it corresponds to a customer arrival, and we must compute the available products. The data used is from Tables C.2 and C.3.

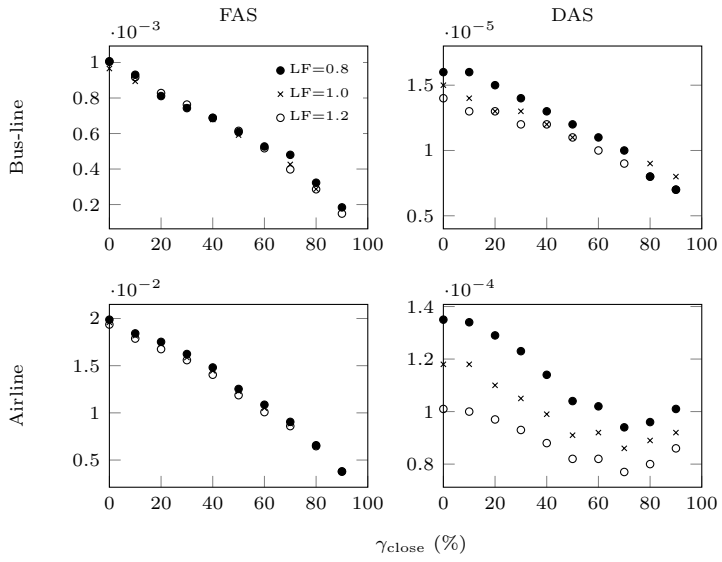

Figure 6.11 Evaluation time CPU/N for the two estimators, averaged over the load factors 0.8, 1.0, and 1.2.

We note that the CRS call time decreases as γ_{close} increases for both estimators. This is because both calculations are easier when fewer products are offered and thus for a higher γ_{close}.

We observe that it takes longer to compute a FAS change than a DAS arrival. On average, it is 56.3 times longer for the bus-line and 141.1 for the airline. This is because determining the next change is more complicated than simply computing the available products. We

must determine for each remaining resource the time step to depletion and for each available product the time step to unavailability. The minimum time step corresponds to the next change. On the other hand, computing the available products involves simply checking if a product is available for the policy and if its resources have remaining capacity.

Note that the results for estimation times depend on how the estimators are coded. We tried to find the best implementation of each approach to give a fair comparison.

In conclusion, the experiments show that FAS outperforms DAS in terms of estimation time. As expected, the time to calculate each change (CRS^F) is greater than the time to compute each arrival (CRS^D). However, the number of arrivals may be large, depending on the desired precision and the number of arrivals per evaluation.

6.5.3 Optimization

We now compare the two estimators in terms of solving the CNRM problem. The goal is to use simulation to converge to the availability policy returning the best expected revenue.

We start by implementing the SP algorithm (Section 6.4.1) for the FAS estimator. We use the parameters $\alpha = 0.01$ and $\alpha = 0.001$ respectively for the random and the optimal starting point. We also set $\beta = 0.5$. In Figure 6.12, we report the convergence of this method (μ^F) for a random and an optimal starting point. The availability policies found during the convergence are simulated by the DAS estimator (μ^D).

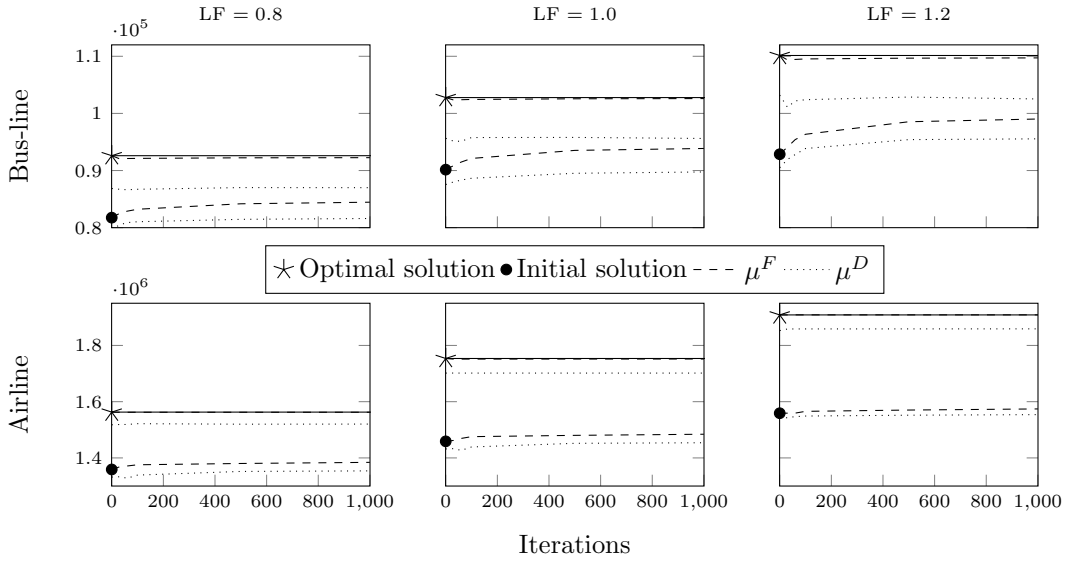

Figure 6.12 SP technique applied to the FAS estimator (μ^F). The solutions returned by FAS are simulated by DAS (μ^D).

The main message from Figure 6.12 is that the SP needs a good starting point to find a near-

optimal solution. We stop the process after 1000 iterations, but the additional improvement is insignificant because of the step size h_j as explained in Section 6.4.1. We adjusted the parameters α and β, but the convergence was worse.

For the optimal starting point, the SP starts by worsening the solution and then converges to near-optimality. This is because the SP leaves the optimal area corresponding to a specific local maximum.

The method performs 1000 iterations in approximately 12 min for the bus-line and between 50 and 120 min for the airline. It is much slower than solving the CDLP, and the solution is at best equivalent.

Note that both estimators have approximately the same shape over the SP convergence. The estimated DAS revenue is lower than that for FAS because of the bias explained in Section 6.3.3 and demonstrated in Section 6.5.1. We note that the bias is approximately constant, and that the optimization on FAS is similarly reflected on DAS.

We conclude that the equivalence with the CDLP makes it difficult for a FAS simulation-based optimization technique to be as efficient as the solution of this mathematical program. Moreover, FAS does not take into account the arrival order stochasticity to improve the robustness of the solutions.

However, the equivalence also allows us to develop methods to support the CNRM optimization. Without going into details, we present two simple examples. The results are reported in Figure 6.13, and the method is explained below.

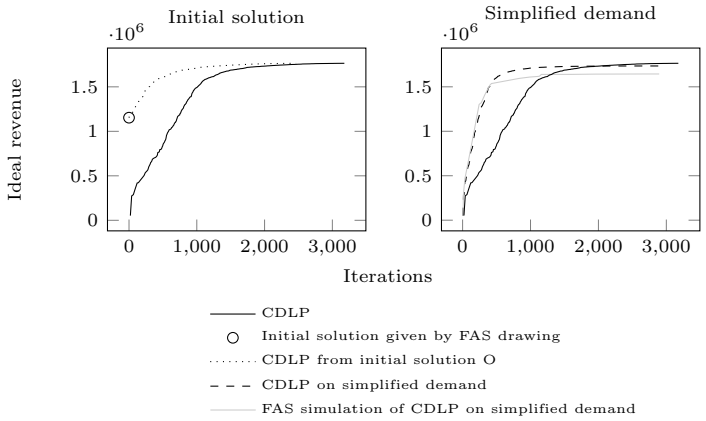

Figure 6.13 Two examples of FAS optimization support in the solution of the CDLP for the airline instance (LF=1).

First, we use FAS to generate a good initial solution for the CDLP. FAS is fast, and we randomly generate as many availability policies as possible in 15 s. The best one is used as the CDLP initial solution. The results are reported on the left of Figure 6.13 for the airline

instance with $LF = 1$. The best availability policy had a revenue of 1154306.21 (indicated by a circle in the graph), which is 34% lower than the optimal solution. With our approach, we save approximately 700 s, which is worthwhile given the total solution time is 2000 s. However, a tabu search or a genetic algorithm might provide a better initial solution.

Second, we use FAS to determine the CDLP revenue given the full demand for availability policies found by a CDLP solved for a partial demand. We remove the product with the lowest probability from each segment consideration set. We wish to focus on the most important component of the demand. The results are reported on the right of Figure 6.13 for the airline instance with $LF = 1$. We observe that the final revenue is approximately 7% lower than that for the CDLP with full demand, but we also save approximately 700 s. We could also use the convergence over partial demand to supply a good initial solution for the CDLP with full demand.

In conclusion, the experiments of this section show that FAS is not necessarily a good estimator for an simulation-based optimization because of its equivalence to the CDLP. However, it is fast, and the equivalence allows it to efficiently support the CNRM optimization. The two approaches tested here could potentially be used in other applications.

6.6 Conclusion

We have proposed a new simulation estimator for CNRM. It estimates the expected revenue of an availability policy by considering fluid arrivals. Requiring only one evaluation, our approach is much faster than the traditional Monte-Carlo simulation based on discrete arrivals. Our estimator is therefore invariant but biased and can underestimate as well as overestimate. The associated bias is almost impossible to measure a priori and can in theory be arbitrarily large. Experiments show that the bias is minimal in practice for a large instance and seems to be asymptotically null. However, it is higher for the optimal availability policy for which the fluid aspect is emphasized. We investigated whether our estimator can solve or support the optimization of the availability policy. Tests on simulation-based optimization methods showed that a good starting point is crucial. We proved that our estimator is equivalent to a widely used approximation for this problem. It is thus preferable to solve the latter rather than use our simulation to converge locally. In particular, our estimator cannot take into account the arrival stochasticity to improve the solution robustness. The equivalence allows us to develop new methods to support the optimization, and we proposed two simple approaches that significantly accelerate the solution of the tested instance. Our estimator is promising because it is fast even for large instances and returns acceptable estimations.

Acknowledgment

The authors are grateful to the Natural Sciences and Engineering Research Council of Canada (NSERC), the Fonds de recherche du Quebec en nature et technologies (FRQNT), and Ex-Pretio technologies for funding and supporting this research.

CHAPITRE 7 DISCUSSION GÉNÉRALE

Dans cette thèse, nous traitons le problème de gestion de la disponibilité (GD) au cours d'une période de réservation où l'on considère plusieurs ressources et une demande modélisée avec comportement d'achat. Nous avons introduit des concepts et mis au point des approches qui permettent d'améliorer l'optimisation et la simulation de ce problème. Ces améliorations touchent à la robustesse des contrôles de disponibilité retournés, au temps de résolution de l'optimisation et de la simulation, à la taille des instances pouvant être résolues et à la généralisation vis à vis des différentes modélisations du comportement d'achat. Les cinq concepts principaux de ce doctorat sont les suivants :

— Contrôle de disponibilité par temps de fermeture produit.

— Résolution via des variables binaires représentant la hiérarchie de vente entre produits.

— Graphe orienté acyclique généralisant les différentes modélisations de comportement d'achat.

— Méthode de raffinement de la demande pour une résolution itérative.

— Simulation à arrivées fluide pour la simulation en GD.

7.1 Synthèse des travaux

Au chapitre 4, nous avons présenté l'approximation *Product Closing Program* (PCP) du problème de gestion de la disponibilité sous comportement d'achat non-paramétrique grâce à un contrôle de disponibilité par temps de fermeture de produit qui est un nouveau type de contrôle pour le domaine de GD. Nous avons expliqué que le PCP n'est pas nécessairement une borne supérieure pour la formulation exacte du problème dans le cas général. Mais que pour le cas de non réouverture, il est équivalent à l'approximation *Choice Determinitic Linear Program* (CDLP) qui est l'approximation de référence. Par héritage des caractéristiques du CDLP, le PCP devient alors une borne supérieure pour la formulation exacte et est asymptotiquement optimale. Notre approximation prend tout son sens car la non réouverture est souvent souhaitée en pratique. Le PCP étant non-linéaire, nous avons proposé une linéarisation en nombres entiers *Product Closing Mixed Program* (PCMP) qui se base sur des variables binaires de hiérarchie de vente entre produits. Celles-ci permettent une modélisation concise réduisant l'aspect combinatoire du modèle. Elles trouvent en outre une signification pratique très utile. Ainsi, nous avons développé un modèle linéaire *Product Closing Linear Program* (PCLP) du PCP lorsque la hiérarchie est fixée apriori. Il est facile en pratique de trouver une

hiérarchie retournant un revenu proche de celui donné par celle optimale. Nous avons ainsi choisi la hiérarchie qui ordonne les produits par prix croissant. Nous avons alors une solution qui est de bonne qualité et qui est rapidement obtenu par un modèle complétement linéaire et continue qu'est le PCLP. Celle-ci peut servir de solution initiale à la résolution du PCMP et donc diminuer grandement son temps de résolution. Finalement on peut déduire des durées d'offre à partir des temps de fermeture. Nous pouvons donc nous servir des contrôles retournés par notre approche comme solution initiale au CDLP. Les résultats numériques, sur des instances correspondant à des problèmes industriels, montrent que PCMP retourne beaucoup plus rapidement une solution que le CDLP. Cette différence en temps de résolution se creuse sur les instances de taille importante. Pour une instance de très grande taille représentant une partie du réseau aérien de Delta Airline, PCMP se résout en moins d'une quinzaine de seconds là où le CDLP nécessite en moyenne plus d'une heure. Les contrôles de disponibilité retournés par notre approche génèrent un revenu espéré, obtenu par une simulation réaliste, légèrement supérieur à la meilleure approximation existante. Le PCMP permet aussi lorsqu'il est choisi comme solution initiale du CDLP, d'accélérer par trois la résolution de celui-ci. Enfin, le PCLP est comme attendu l'approximation qui se résout le plus rapidement et a un revenu espéré seulement 0.6% inférieur à celui du CDLP.

Au chapitre 5, nous avons généralisé l'approximation PCP précédente à n'importe quel comportement d'achat. Pour ce faire, nous avons montré que tout comportement d'achat peut se mettre sous la forme de chemins d'achat formant un graphe orienté acyclique. Ce dernier possède une structure non paramétrique qui permet d'étendre les idées du PCP. Nous avons alors obtenu l'approximation non linéaire *Buying Graph Program* (BGP) qui partage les mêmes caractéristiques que le PCP au niveau temps de fermeture et réouverture. Nous avons montré que les résultats théoriques sur le PCP restent valides pour le BGP que ce soit pour la borne supérieure de la formulation exacte ou pour l'optimalité asymptotique. Nous avons linéarisé le BGP en *Buying Graph Mixed Program* (BGMP) en utilisant les mêmes variables binaires de hiérarchie que précédemment. Pour une hiérarchie fixée, nous avons alors formulé le modèle linéaire *Buying Graph Linear Program* (BGLP) qui donne le même résultat que BGMP lorsque la hiérarchie optimale est utilisée. Les résultats numériques sur des instances de la littérature montrent que notre approximation performe au moins aussi bien que celles traditionnelles. Ces instances restent relativement de petite et moyenne taille. Par rapport au comportement non paramétrique, la modélisation paramétrique peut faire exploser la taille du graphe de demande si le nombre de produit par segment augmente ou si l'instance est de taille importante. Pour pallier à cela, nous avons développé une méthode de résolution qui permet de résoudre le BGMP itérativement. Cette méthode ajoute progressivement des chemins d'achat dans le but de raffiner petit à petit la modélisation de la demande. Nous

avons proposé plusieurs stratégies d'ajout de chemins d'achat successifs. Le but étant de capturer les principales tendances et logiques de cette demande le plus rapidement possible. Les résultats sur les instances de grande taille montrent l'excellente performance de notre méthode itérative de résolution qui permet à BGMP de converger vers une bonne solution beaucoup plus rapidement que les approches existantes par génération de colonnes. Cette bonne solution est souvent obtenue après des heures de convergence pour les approximations traditionnelles alors que notre approche l'atteint en quelques minutes. De plus ces expériences montrent que notre approche retourne très rapidement une solution très proche de l'optimale et donc que la plus grosse partie de la convergence est déjà effectuée.

Au chapitre 6 nous avons introduit un nouvel estimateur *Fluid Arrivals Simulation* (FAS) basé sur le concept d'arrivées fluides. Nous avons montré, qu'en agrégeant les différentes arrivées par segment, le FAS retourne un revenu espéré uniquement en une évaluation et est donc invariant contrairement au modèle traditionnel d'arrivées discrètes *Discrete Arrivals Simulation* (DAS). Cependant, ce revenu présente un biais pouvant être en théorie arbitrairement grand alors que l'estimateur DAS est non biaisé. Nous avons alors expliqué que ce biais est lié à des aspects discrets qui ont une influence minime sur les instances que l'on retrouve en pratique. Des expériences numériques sur des instances de grandes tailles témoignent de la nette supériorité de notre estimateur en termes de temps de calcul ainsi que d'un faible biais comme attendu. Ce gain important de vitesse d'estimation est primordial pour de nombreuses applications où il faut retourner rapidement un estimé du revenu à espérer : optimisation basée sur la simulation, validation de prédiction de la demande et approches "what if" par exemple. Nous avons aussi montré que le revenu estimé par le FAS est équivalent à la plupart des approximations d'optimisation en gestion de disponibilité. Notre estimateur peut alors être utilisé pour appuyer ces optimiseurs et accélérer leur résolution.

7.2 Limites des solutions proposées et améliorations futures

L'approximation PCMP est parfaitement adaptée à la demande non paramétrique. Les différentes techniques et modélisations développées ont permis d'atteindre une résolution très rapide même pour des instances de taille importante. Le modèle PCLP est encore plus efficace lorsque la taille des instances augmente davantage car il ne souffre pas des multiples contraintes modélisant la hiérarchie entre la fermeture des produits et est un modèle continue. Ce dernier peut retourner un contrôle de disponibilité qui génère un revenu proche de celui de la solution optimale du PCMP si la hiérarchie est bien choisie. Cette approche peut être une solution à la résolution des plus grosses instances. La recherche de la hiérarchie optimale peut être sujet à de nombreux développements ou heuristiques. Dans ce doctorat

nous prenons la simple heuristique de classer les produits par ordre croissant de revenu. Dans ce cas-ci, nous ne considérons pas le nombre de ressources utilisées ou la demande pour chacun des produits. Une heuristique plus fine pourrait donc facilement être développée. Les pistes que nous pourrions aussi explorer sont des algorithmes génétiques et des méthodes d'apprentissage par renforcement.

La résolution itérative du modèle de BGMP assure une convergence très rapide vers une solution correspondant seulement à une partie du comportement de la demande. Les stratégies d'ajout de chemins d'achats proposées permettent un raffinement de cette demande au fur et à mesure des ajouts de choix. Pour les premières itérations, les solutions retournées présentent alors un très bon revenu idéal mais le revenu espéré correspondant est mauvais. En effet, les choix omis au début dans l'optimisation se révèlent avoir en fait une grande influence lorsque la demande complète est considérée comme c'est le cas dans la simulation. Ceci est alors corrigé au fur et à mesure que l'arbre se construit mais reste trompeur sur la qualité réelle du revenu idéal à chaque itération de résolution. Nous pourrions alors repenser la modélisation et/ou la stratégie d'ajout de façon à restreindre immédiatement ou progressivement l'influence de la partie de demande qui n'est pas encore ajoutée. Il s'agit par exemple d'ajouter uniquement les chemins d'achats qui sont susceptibles de changer la solution actuelle. On pourrait aussi penser ajouter des contraintes rendant compte de l'impact des choix qu'il reste à rajouter.

Nous avons introduit l'estimateur FAS pour la simulation. Cet estimateur a la propriété de retourner un revenu espéré égal au revenu idéal des principales approximations statiques pour l'optimisation de la gestion de la disponibilité. De nombreux travaux peuvent alors être menés pour se servir de cet estimateur comme support à l'optimisation. Nous aurions par exemple voulu tester le FAS pour déterminer le vrai revenu idéal du BGMP lors de la méthode itérative. Nous aurions pu aussi étudier plus en détail cet estimateur pour déterminer le gradient correspondant et ainsi élaborer une méthode d'optimisation basée sur la simulation propre a cet estimateur. La contrepartie du temps d'évaluation pour le FAS est le biais qu'il peut engendrer. On observe que celui-ci est en pratique très faible mais il faudrait pouvoir le quantifier au préalable pour chaque instance. Une étude d'analyse permettrait sûrement d'établir des bornes sur ce biais.

CHAPITRE 8 CONCLUSION ET RECOMMENDATIONS

En conclusion, cette thèse se bâtie autour de cinq nouveaux concepts. Le premier est un nouveau contrôle de disponibilité par temps de fermeture de vente de produits au cours de la période de réservation. Celui-ci est plus adapté à des modélisations de comportement d'achat non paramétrique par rapport à ceux traditionnels par durée d'offres. Le deuxième est l'intégration de variables binaires de hiérarchie de fermeture entre produit. Celles-ci ont un réel sens pratique et permettent de développer plus facilement des heuristiques pour accélérer la résolution de nos modèles. Le troisième est la généralisation des comportements d'achat à un graphe orienté acyclique. Celui-ci a une structure non paramétrique même si la modélisation initiale est paramétrique. Les approches développées pour le non paramétrique peuvent alors s'appliquer à n'importe quelle modélisation du comportement d'achat. Le quatrième est la méthode de résolution itérative construisant progressivement le graphe de demande précédent. Cette méthode permet de converger beaucoup plus rapidement que les approches de génération de colonnes traditionnelles vers une solution proche de l'optimalité grâce à un raffinement progressif de la demande. Enfin le cinquième est la simulation par arrivées fluides plutôt que discrètes comme usuellement en gestion de disponibilité. Ce type de simulation évalue beaucoup plus rapidement une approximation du revenu à espérer. Le biais engendré est minime en pratique. Ces cinq concepts ont permis d'établir de nouveaux modèles et méthodes qui améliorent grandement la recherche du contrôle optimal et l'évaluation de celui-ci pour le problème de gestion de la disponibilité. Nous obtenons alors des meilleurs temps de d'optimisation et d'évaluation, des solutions plus robustes, des gains en revenu à espérer et la possibilité de résoudre des instances encore plus importantes en taille. L'industrie devant résoudre ce problème dans une fenêtre de temps allouée et pour des revenus en jeux élevés, nos approches promettent d'excellentes retombées pratiques et financières.

RÉFÉRENCES

Daniel Adelman (2007). Dynamic bid prices in revenue management. *Operations Research*, *55*(4), 647–661.

Belobaba, Peter (1987a). Air travel demand and airline seat inventory management. Rapport technique, Cambridge, MA : Flight Transportation Laboratory, Massachusetts Institute of Technology,[1987].

Belobaba, Peter (1992). Optimal vs. heuristic methods for nested seat allocation. *Presentation at ORSA/TIMS Joint National Meeting.*

Belobaba, Peter (2001). Revenue and competitive impacts of od control : Summary of pods results. *First Annual INFORMS Revenue Management Section Meeting. New York, NY.*

Belobaba, PP and Hopperstad, C (1999). Boeing/mit simulation study : Pods results update. *1999 AGIFORS Reservations and Yield Management Study.* MIT simulation study : PODS results update. In 1999 AGIFORS Reservations and Yield Management Study Group Symposium, April.

Peter P. Belobaba (1987b). Survey paper—airline yield management an overview of seat inventory control. *Transportation Science*, *21*(2), 63–73.

Peter P. Belobaba (1989). OR practice—application of a probabilistic decision model to airline seat inventory control. *Operations Research*, *37*(2), 183–197.

Moshe Ben-Akiva and Steven R. Lerman (1985). *Discrete Choice Analysis : Theory and Application to Travel Demand.*, vol. 9. MIT Press.

Dimitris Bertsimas and Sanne de Boer (2005a). Simulation-based booking limits for airline revenue management. *Operations Research*, *53*(1), 90–106.

Dimitris Bertsimas and Sanne de Boer (2005b). Special issue papers : Dynamic pricing and inventory control for multiple products. *Journal of Revenue and Pricing Management*, *3*(4), 303–319.

Dimitris Bertsimas and Ioana Popescu (2003). Revenue management in a dynamic network environment. *Transportation Science*, *37*(3), 257–277.

Marco Bijvank and Pierre L'Ecuyer and Patrice Marcotte (2011). Rmsim : A java library for simulating revenue management systems. *Proceedings of the Winter Simulation Conference.* Winter Simulation Conference, 2703–2714.

blinded for peer review (2018). Product-closing for nonparametric choice network revenue management. *Submitted for publication.*

Juan José Miranda Bront and Isabel Méndez-Díaz and Gustavo Vulcano (2009). A column generation algorithm for choice-based network revenue management. *Operations Research*, *57*(3), 769–784.

S. L. Brumelle and J. I. McGill (1993). Airline seat allocation with multiple nested fare classes. *Operations Research*, *41*(1), 127–137.

Carrier, Emmanuel (2003). *Modeling airline passenger choice : Passenger preference for schedule in the passenger origin-destination simulator (PODS)*. Thèse de doctorat, Massachusetts Institute of Technology.

Emmanuel Carrier and Larry Weatherford (2015). Implementation of mnl choice models in the passenger origin-destination simulator (pods). *Journal of Revenue and Pricing Management*, *14*(6), 400–407.

Juan M. Chaneton and Gustavo Vulcano (2011). Computing bid prices for revenue management under customer choice behavior. *Manufacturing and Service Operations Management*, *13*(4), 452–470.

Lijian Chen and Tito Homem-de-Mello (2009). Re-solving stochastic programming models for airline revenue management. *Annals of Operations Research*, *177*(1), 91–114.

Lijian Chen and Tito Homem-de-Mello (2010). Mathematical programming models for revenue management under customer choice. *European Journal of Operational Research*, *203*(2), 294–305.

A. Ciancimino and G. Inzerillo and S. Lucidi and L. Palagi (1999). A mathematical programming approach for the solution of the railway yield management problem. *Transportation Science*, *33*(2), 168–181.

Catherine Cleophas and Michael Frank and Natalia Kliewer (2009). Simulation-based key performance indicators for evaluating the quality of airline demand forecasting. *Journal of Revenue and Pricing Management*, *8*(4), 330–342.

William L. Cooper (2002). Asymptotic behavior of an allocation policy for revenue management. *Operations Research*, *50*(4), 720–727.

Darot, Jérémy François Jean (2001). *Revenue management for airline alliances : Passenger origin-destination simulation analysis*. Thèse de doctorat, Massachusetts Institute of Technology.

Sanne V. de Boer and Richard Freling and Nanda Piersma (2002). Mathematical programming for network revenue management revisited. *European Journal of Operational Research*, *137*(1), 72–92.

DeMiguel, Victor and Mishra, Nishant (2006). What multistage stochastic programming can do for network revenue management. Rapport technique, Working paper, London Business School, 2006 and Optimization Online URL : http ://www. optimization-online. org/DB HTML/2006/10/1508. html.

Goda R. Doreswamy and Aditya S. Kothari and Sumala Tirumalachetty (2015). Simulating the flavors of revenue management for airlines. *Journal of Revenue and Pricing Management*, *14*(6), 421–432.

D'sylva, E (1982). O and d seat assignment to maximize expected revenue. *Unpublished internal report, Boeing Commercial Airplane Company, Seattle, WA*.

Eguchi, Takeshi and Belobaba, Peter P (2004). Modelling and simulation of impact of revenue management on japan's domestic market. *Journal of Revenue and Pricing Management*, *3*(2), 119–142.

Alexander Erdelyi and Huseyin Topaloglu (2010). Using decomposition methods to solve pricing problems in network revenue management. *Journal of Revenue and Pricing Management*, *10*(4), 325–343.

Vivek F. Farias and Srikanth Jagabathula and Devavrat Shah (2013). A nonparametric approach to modeling choice with limited data. *Management Science*, *59*(2), 305–322.

Daniel R. Figueiredo and Benyuan Liu and Yang Guo and Jim Kurose and Don Towsley (2006). On the efficiency of fluid simulation of networks. *Computer Networks*, *50*(12), 1974–1994.

Fiig, Thomas and Härdling, Roger and Pölt, Stefan and Hopperstad, Craig (2014). Demand forecasting and measuring forecast accuracy in general fare structures. *Journal of Revenue and Pricing Management*, *13*(6), 413–439.

Michael Frank and Martin Friedemann and Michael Mederer and Anika Schroeder (2006). Airline revenue management : A simulation of dynamic capacity management. *Journal of Revenue and Pricing Management*, *5*(1), 62–71.

Michael Frank and Martin Friedemann and Anika Schröder (2008). Principles for simulations in revenue management. *Journal of Revenue and Pricing Management*, *7*(1), 7–16.

Gallego, Guillermo and Iyengar, Garud and Phillips, R and Dubey, Abha (2004). Managing flexible products on a network. Rapport technique TR-2004-01, Columbia University.

Gallego, Guillermo and Ratliff, Richard and Shebalov, Sergey (2011). A general attraction model and an efficient formulation for the network revenue management problem. *Technical Report, Columbia University, New York, NY*.

Gallego, Guillermo and Ratliff, Richard and Shebalov, Sergey (2014). A general attraction model and sales-based linear program for network revenue management under customer choice. *Operations Research*, *63*(1), 212–232.

Guillermo Gallego and Garrett van Ryzin (1997). A multiproduct dynamic pricing problem and its applications to network yield management. *Operations Research*, *45*(1), 24–41.

Gilks, Walter R and Richardson, Sylvia and Spiegelhalter, David (1995). *Markov chain Monte Carlo in practice*. CRC press.

Fred Glover and Randy Glover and Joe Lorenzo and Claude McMillan (1982). The passenger-mix problem in the scheduled airlines. *Interfaces*, *12*(3), 73–80.

Gorin, Thomas and Belobaba, Peter P (2004). Special issue papers : Revenue management performance in a low-fare airline environment : Insights from the passenger origin—destination simulator. *Journal of Revenue and Pricing Management*, *3*(3), 215–236.

Abhijit Gosavi (2015). Simulation-based optimization.

Abhijit Gosavi and Emrah Ozkaya and Aykut F. Kahraman (2005). Simulation optimization for revenue management of airlines with cancellations and overbooking. *OR Spectrum*, *29*(1), 21–38.

Ward Hanson and Kipp Martin (1996). Optimizing multinomial logit profit functions. *Management Science*, *42*(7), 992–1003.

Hosseinalifam, Morad and Marcotte, Patrice and Savard,Gilles (2014). *A mathematical programming framework for network capacity control in customer choice-based revenue managment*. Thèse de doctorat, École polytechnique Montréal.

M. Hosseinalifam and P. Marcotte and G. Savard (2015). Network capacity control under a nonparametric demand choice model. *Operations Research Letters*, *43*(5), 461–466.

Morad Hosseinalifam and Gilles Savard and Patrice Marcotte (2016). Computing booking limits under a non-parametric demand model : A mathematical programming approach. *Journal of Revenue and Pricing Management*, *15*, 170–184.

Srikanth Jagabathula and Paat Rusmevichientong (2017). A nonparametric joint assortment and price choice model. *Management Science*, *63*(9), 3128–3145.

Jasin, Stefanus and Kumar, Sunil (2012). A re-solving heuristic with bounded revenue loss for network revenue management with customer choice. *Mathematics of Operations Research*, *37*(2), 313–345.

Kesidis, G and Singh, A and Cheung, D and Kwok, WW (1996). Feasibility of fluid event-driven simulation for atm networks. *Global Telecommunications Conference, 1996.* IEEE, vol. 3, 2013–2017.

Sumit Kunnumkal and Huseyin Topaloglu (2008). A refined deterministic linear program for the network revenue management problem with customer choice behavior. *Naval Research Logistics (NRL)*, *55*(6), 563–580.

Sumit Kunnumkal and Huseyin Topaloglu (2010). A new dynamic programming decomposition method for the network revenue management problem with customer choice behavior. *Production and Operations Management*, *19*(5), 575–590.

Kunnumkal, Sumit and Topaloglu, Huseyin (2011). A randomized linear program for the network revenue management problem with customer choice behavior. *Journal of Revenue and Pricing management*, *10*(5), 455–470.

Conrad J. Lautenbacher and Shaler Stidham (1999). The underlying markov decision process in the single-leg airline yield-management problem. *Transportation Science*, *33*(2), 136–146.

Littlewood, Kenneth (1972). Forecasting and control of passenger bookings. *Airline Group International Federation of Operational Research Societies Proceedings, 1972*, *12*, 95–117.

Qian Liu and Garrett van Ryzin (2008). On the choice-based linear programming model for network revenue management. *Manufacturing and Service Operations Management*, *10*(2), 288–310.

Jeffrey I. McGill and Garrett J. van Ryzin (1999). Revenue management : Research overview and prospects. *Transportation Science*, *33*(2), 233–256.

Joern Meissner and Arne Strauss (2012). Network revenue management with inventory-sensitive bid prices and customer choice. *European Journal of Operational Research*, *216*(2), 459–468.

Joern Meissner and Arne Strauss and Kalyan Talluri (2013). An enhanced concave program relaxation for choice network revenue management. *Production and Operations Management*, *22*(1), 71–87.

Richard M. Ratliff and B. Venkateshwara Rao and Chittur P. Narayan and Kartik Yellepeddi (2008). A multi-flight recapture heuristic for estimating unconstrained demand from airline bookings. *Journal of Revenue and Pricing Management*, *7*(2), 153–171.

Lawrence W. Robinson (1995). Optimal and approximate control policies for airline booking with sequential nonmonotonic fare classes. *Operations Research*, *43*(2), 252–263.

Paat Rusmevichientong and Benjamin Van Roy and Peter W. Glynn (2006). A nonparametric approach to multiproduct pricing. *Operations Research*, *54*(1), 82–98.

Paat Rusmevichientong and David Shmoys and Chaoxu Tong and Huseyin Topaloglu (2014). Assortment optimization under the multinomial logit model with random choice parameters. *Production and Operations Management*, *23*(11), 2023–2039.

Simpson, Robert Warren (1989). *Using network flow techniques to find shadow prices for market demands and seat inventory control*. MIT, Department of Aeronautics and Astronautics, Flight Transportation Laboratory.

Smith, BC and Penn, CW (1988). Analysis of alternate origin-destination control strategies. *AGIFORS PROCEEDINGS*.

Spall, James C (1998). An overview of the simultaneous perturbation method for efficient optimization. *Johns Hopkins apl technical digest*, *19*(4), 482–492.

Arne K. Strauss and Robert Klein and Claudius Steinhardt (2018). A review of choice-based revenue management : Theory and methods. *European Journal of Operational Research*.

Arne K. Strauss and Kalyan Talluri (2017). Tractable consideration set structures for assortment optimization and network revenue management. *Production and Operations Management*, *26*, 1359–1368.

Talluri, Kalyan (2010). A randomized concave programming method for choice network revenue management. *Working Papers (Departamento de Economía y Empresa, Universitat Pompeu Fabra)*.

Kalyan Talluri (2014). New formulations for choice network revenue management. *INFORMS Journal on Computing*, *26*(2), 401–413.

Kalyan Talluri and Garrett van Ryzin (1998). An analysis of bid-price controls for network revenue management. *Management Science*, *44*(11-part-1), 1577–1593.

Kalyan Talluri and Garrett van Ryzin (1999). A randomized linear programming method for computing network bid prices. *Transportation Science*, *33*(2), 207–216.

Kalyan Talluri and Garrett van Ryzin (2004a). Revenue management under a general discrete choice model of consumer behavior. *Management Science*, *50*(1), 15–33.

Kalyan Talluri and Garrett van Ryzin (2004b). *The Theory and Practice of Revenue Management*, vol. 68. Springer Science and Business Media.

Huseyin Topaloglu (2008). A stochastic approximation method to compute bid prices in network revenue management problems. *INFORMS Journal on Computing*, *20*(4), 596–610.

Garrett van Ryzin and Gustavo Vulcano (2008a). Computing virtual nesting controls for network revenue management under customer choice behavior. *Manufacturing and Service Operations Management*, *10*(3), 448–467.

Garrett van Ryzin and Gustavo Vulcano (2008b). Simulation-based optimization of virtual nesting controls for network revenue management. *Operations Research*, *56*(4), 865–880.

Garrett van Ryzin and Gustavo Vulcano (2015). A market discovery algorithm to estimate a general class of nonparametric choice models. *Management Science*, *61*(2), 281–300.

Garrett van Ryzin and Gustavo Vulcano (2017). Technical note—an expectation-maximization method to estimate a rank-based choice model of demand. *Operations Research*, *65*(2), 396–407.

Weatherford, Larry R (2013). Improved revenues from various unconstraining methods in a passenger origin-destination simulator (pods) environment with semi-restricted fares. *Journal of Revenue and Pricing Management*, *12*(1), 60–82.

Williamson, Elizabeth Louise (1992). Airline network seat inventory control : Methodologies and revenue impacts. Rapport technique, [Cambridge, Mass. : Massachusetts Institute of Technology, Dept. of Aeronautics &amp ; Astronautics], Flight Transportation Laboratory,[1992].

Dan Zhang (2011). An improved dynamic programming decomposition approach for network revenue management. *Manufacturing and Service Operations Management*, *13*(1), 35–52.

Dan Zhang and Daniel Adelman (2009). An approximate dynamic programming approach to network revenue management with customer choice. *Transportation Science*, *43*(3), 381–394.

Dan Zhang and Larry Weatherford (2017). Dynamic pricing for network revenue management : A new approach and application in the hotel industry. *INFORMS Journal on Computing*, *29*(1), 18–35.

ANNEXE A ARTICLE 1 E-COMPANION

A.1 Instances

Tableau A.1 Instances characteristics. [1] five cities. We use the preference list (PL) choice behavior as presented in the article.

	Parallel flights	Bus-line	Airline[1]
Number of resources	3	6	115
Number of journeys	3	2	115
Number of Markets	1	6	20
Number of products	6	24	1591
Number of segments	4	18	438
Choice behaviors	PL	PL	PL
Largest consideration set	6	4	10
Smallest consideration set	2	4	1
Average consideration set	4.00	4.00	7.93

Instances are entirely described in CSV files at :

http ://thibaultbarbier.com

A.2 Parallel flights

Tableau A.2 Parallel flights results

Approx >		CDLP			SDCP		PCMP		
Policy >		OP	PB	OD	PB	OD	PC	PB	OD
0.6	CPUa		0.23		0.05			0.03	
	CPUp	0.00	0.00	4.27	0.00	4.16	0.02	0.00	4.14
	CPU	0.23	0.23	4.50	0.05	4.21	0.05	0.03	4.17
	R		23943.4		23943.4			23943.4	
	E[R]	23918.6±0.28%	22995.2±0.18%	23925.5±0.28%	22997.4±0.17%	23915.8±0.28%	23887.8±0.28%	23003.3±0.17%	23904.1±0.28%
	ΔE[R]	–	-3.86	0.03	-3.85	-0.01	-0.13	-3.83	-0.06
	CF		0.5321		0.5321			0.5321	
	E[CF]	0.5315±0.28%	0.5110±0.18%	0.5317±0.28%	0.5111±0.17%	0.5315±0.28%	0.5308±0.28%	0.5112±0.17%	0.5312±0.28%
	ΔE[CF]	–	-3.86	0.03	-3.85	-0.01	-0.13	-3.83	-0.06
0.8	CPUa		0.03		0.03			0.03	
	CPUp	0.00	0.00	6.27	0.00	6.52	0.00	0.00	6.68
	CPU	0.03	0.03	6.30	0.03	6.55	0.03	0.03	6.71
	R		31539.6		31539.6			31539.6	
	E[R]	31206.1±0.23%	30780.4±0.15%	31378.3±0.22%	30721.4±0.14%	31393.1±0.21%	30418.9±0.15%	30686.9±0.14%	31424.4±0.22%
	ΔE[R]		-1.36	0.55	-1.55	0.60	-2.52	-1.66	0.70
	CF		0.7009		0.7009			0.7009	
	E[CF]	0.6935±0.23%	0.6840±0.15%	0.6973±0.22%	0.6827±0.14%	0.6976±0.21%	0.6760±0.15%	0.6819±0.14%	0.6983±0.22%
	ΔE[CF]	–	-1.36	0.55	-1.55	0.60	-2.52	-1.66	0.70
1.0	CPUa		0.55		0.03			0.02	
	CPUp	0.00	0.00	8.55	0.00	8.57	0.00	0.00	8.85
	CPU	0.55	0.55	9.10	0.03	8.60	0.02	0.02	8.87
	R		37924.5		37924.5			37924.5	
	E[R]	37080.4±0.17%	37096.5±0.11%	37928.9±0.20%	37142.2±0.12%	37873.3±0.19%	37793.1±0.21%	37185.9±0.12%	37903.3±0.20%
	ΔE[R]	–	0.04	2.29	0.17	2.14	1.92	0.28	2.22
	CF		0.84		0.84			0.84	
	E[CF]	0.8240±0.17%	0.8244±0.11%	0.8429±0.20%	0.8254±0.12%	0.8416±0.19%	0.8398±0.21%	0.8264±0.12%	0.8423±0.20%
	ΔE[CF]	–	0.04	2.29	0.17	2.14	1.92	0.28	2.22
1.2	CPUa		0.27		0.2			0.02	
	CPUp	0.00	0.00	10.68	0.00	10.76	0.00	0.00	10.78
	CPU	0.27	0.27	10.95	0.02	10.78	0.02	0.02	10.80
	R		44309.4		44309.4			44309.4	
	E[R]	43237.7±0.14%	43393.9±0.11%	43589.7±0.13%	43428.0±0.11%	43691.4±0.12%	43416.0±0.14%	43393.4±0.11%	43542.9±0.13%
	ΔE[R]	–	0.36	0.81	0.44	1.05	0.41	0.36	0.71
	CF		0.9847		0.9847			0.9847	
	E[CF]	0.9608±0.14%	0.9643±0.11%	0.9687±0.13%	0.9651±0.11%	0.9709±0.12%	0.9648±0.14%	0.9643±0.11%	0.9676±0.13%
	ΔE[CF]	–	0.36	0.81	0.44	1.05	0.41	0.36	0.71
1.4	CPUa		0.03		0.02			0.04	
	CPUp	0.00	0.00	12.82	0.00	13.09	0.00	0.00	13.09
	CPU	0.03	0.03	12.85	0.02	13.11	0.04	0.04	13.13
	R		45000.0		45000.0			45000.0	
	E[R]	43481.3±0.11%	44990.5±0.01%	44930.0±0.02%	44987.4±0.01%	44942.2±0.01%	43837.1±0.11%	44991.4±0.01%	44990.7±0.01%
	ΔE[R]	–	3.47	3.33	3.46	3.36	0.82	3.47	3.47
	CF		1.0000		1.0000			1.0000	
	E[CF]	0.9663±0.11%	0.9998±0.01%	0.9984±0.02%	0.9997±0.01%	0.9987±0.01%	0.9742±0.11%	0.9998±0.01%	0.9998±0.01%
	ΔE[CF]	–	3.47	3.33	3.46	3.36	0.82	3.47	3.47
$\overline{\text{CPU}}$		0.22	0.22	8.74	0.03	8.65	0.03	0.03	8.74
$\overline{\Delta\text{E[R]}}$		–	-0.27	1.40	-0.27	1.43	0.10	-0.28	1.41
$\overline{\Delta\text{E[CF]}}$		–	-0.27	1.40	-0.27	1.43	0.10	-0.28	1.41

Each approximation is solved in **CPUa** seconds and return an optimal revenue **R** corresponding to a capacity factor **CF**. We then transform this solution to policy(ies). This transformation takes **CPUp** seconds and is then simulated in a discrete arrivals simulation with **3000** evaluations to obtain an expected revenue **E[R]** and expected capacity factor **E[CF]** for a 95% confidence interval. The total running time is **CPU** and we calculate **ΔE[CF]** and **ΔE[R]** the capacity factor and expected revenue relative difference with respect to CDLP-OP.

A.3 Bus-line

Tableau A.3 Bus-line results

Approx >		CDLP			SDCP		PCMP		
Policy >		OP	PB	OD	PB	OD	PC	PB	OD
0.6	CPUa		0.89		0.08			0.05	
	CPUp	0.00	0.00	638.12	0.00	701.34	0.00	0.00	565.74
	CPU	0.89	0.89	639.01	0.08	701.42	0.05	0.05	565.79
	R		13151.6		13151.6			13151.6	
	E[R]	12412.6±0.40%	12039.5±0.43%	12451.6±0.41%	12024.8±0.43%	12460.0±0.38%	12467.8±0.44%	12004.2±0.41%	12442.9±0.39%
	ΔE[R]	–	-3.01	0.31	-3.12	0.38	0.44	-3.29	0.24
	CF		0.8723		0.8723			0.8723	
	E[CF]	0.8182±0.41%	0.8010±0.43%	0.8304±0.42%	0.7991±0.44%	0.8303±0.39%	0.8183±0.43%	0.7978±0.42%	0.8336±0.39%
	ΔE[CF]	–	-2.11	1.48	-2.34	1.47	0.01	-2.49	1.87
0.8	CPUa		1.16		0.09			0.06	
	CPUp	0.00	0.00	935.28	0.00	993.93	0.00	0.00	857.53
	CPU	1.16	1.16	936.45	0.09	994.02	0.06	0.06	857.59
	R		15982.4		15982.4			15982.4	
	E[R]	14908.6±0.39%	14667.6±0.35%	15052.2±0.33%	14678.0±0.34%	15074.7±0.33%	14958.0±0.39%	14654.8±0.34%	15037.9±0.34%
	ΔE[R]	–	-1.62	0.96	-1.55	1.11	0.33	-1.70	0.87
	CF		0.9330		0.9330			0.9330	
	E[CF]	0.8713±0.40%	0.8635±0.33%	0.8945±0.34%	0.8637±0.33%	0.8966±0.33%	0.8756±0.39%	0.8624±0.32%	0.8928±0.34%
	ΔE[CF]	–	-0.90	2.66	-0.86	2.91	0.50	-1.02	2.48
1.0	CPUa		0.74		0.08			0.05	
	CPUp	0.00	0.00	793.64	0.00	811.28	0.00	0.00	795.50
	CPU	0.74	0.74	794.38	0.08	811.36	0.05	0.05	795.55
	R		17896.3		17896.3			17896.3	
	E[R]	16496.3±0.36%	16527.6±0.32%	16971.4±0.31%	16542.7±0.32%	16964.7±0.31%	16736.3±0.37%	16491.7±0.33%	16979.2±0.31%
	ΔE[R]	–	0.19	2.88	0.28	2.84	1.45	-0.03	2.92
	CF		0.9385		0.9385			0.9385	
	E[CF]	0.8622±0.38%	0.8668±0.34%	0.9028±0.33%	0.8683±0.34%	0.9035±0.32%	0.8738±0.39%	0.8662±0.34%	0.9040±0.31%
	ΔE[CF]	–	0.53	4.71	0.71	4.79	1.35	0.46	4.85
1.2	CPUa		0.89		0.09			0.05	
	CPUp	0.00	0.00	1007.49	0.00	882.58	0.00	0.00	866.62
	CPU	0.89	0.89	1008.38	0.09	882.67	0.05	0.05	866.67
	R		19049.7		19049.7			19049.7	
	E[R]	17515.7±0.35%	17882.5±0.31%	18119.9±0.24%	17879.3±0.30%	18185.9±0.23%	17746.6±0.37%	17853.8±0.29%	18096.7±0.24%
	ΔE[R]	–	2.09	3.45	2.08	3.83	1.32	1.93	3.32
	CF		0.9838		0.9838			0.9838	
	E[CF]	0.9012±0.37%	0.9205±0.32%	0.9388±0.25%	0.9206±0.32%	0.9405±0.24%	0.9089±0.37%	0.9190±0.31%	0.9376±0.25%
	ΔE[CF]	–	2.14	4.18	2.16	4.37	0.85	1.98	4.04
1.4	CPUa		0.95		0.10			0.03	
	CPUp	0.00	0.00	1318.78	0.00	1038.65	0.00	0.00	992.59
	CPU	0.95	0.95	1319.33	0.10	1038.75	0.03	0.03	992.62
	R		19988.7		19988.7			19988.7	
	E[R]	18179.3±0.35%	18848.0±0.28%	19113.9±0.25%	18857.8±0.29%	19132.5±0.25%	18581.8±0.35%	18838.9±0.28%	19103.7±0.26%
	ΔE[R]	–	3.68	5.14	3.73	5.24	2.21	3.63	5.08
	CF		1.0000		1.0000			1.0000	
	E[CF]	0.9046±0.35%	0.9373±0.30%	0.9508±0.25%	0.9381±0.30%	0.9518±0.25%	0.9202±0.33%	0.9371±0.29%	0.9489±0.25%
	ΔE[CF]	–	3.61	5.11	3.71	5.22	1.73	3.60	4.90
$\overline{\text{CPU}}$		0.92	0.92	939.51	0.09	885.64	0.05	0.05	5.52
$\overline{\Delta\text{E[R]}}$		–	0.27	2.55	0.28	2.68	1.15	0.11	2.49
$\overline{\Delta\text{E[CF]}}$		–	0.65	3.63	0.68	3.77	0.89	0.51	3.63

Each approximation is solved in **CPUa** seconds and return an optimal revenue **R** corresponding to a capacity factor **CF**. We then transform this solution to policy(ies). This transformation takes **CPUp** seconds and is then simulated in a discrete arrivals simulation with **1000** evaluations to obtain an expected revenue **E[R]** and expected capacity factor **E[CF]** for a 95% confidence interval. The total running time is **CPU** and we calculate **ΔE[CF]** and **ΔE[R]** the capacity factor and expected revenue relative difference with respect to CDLP-OP.

A.4 Airline

Tableau A.4 Bus-line results

	Approx >	CDLP		PCMP		CDPC		PCLP	
	Policy >	OP	PB	PC	PB	OP	PB	PC	PB
0.6	CPUa	5054.15		5.84		1652.13		3.84	
	CPUp	0.02	0.59	0.00	0.02	0.00	0.64	0.02	0.00
	CPU	5054.17	5054.74	5.84	5.86	1652.13	1652.87	3.86	3.84
	R	1322319.0		1321425.5		1322319.0		1308717.1	
	ΔR	–		-0.07		0.00		-1.03	
	E[R]	1286871.1±0.07%	1255551.9±0.06%	1290783.2±0.08%	1255436.7±0.06%	1286643.2±0.07%	1255663.2±0.06%	1277509.3±0.07%	1244115.3±0.06%
	ΔE[R]	–	-2.43	0.30	-2.44	-0.02	-2.43	-0.73	-3.32
	CF	0.7160		0.7188		0.7160		0.7074	
	ΔCF	–		0.40		0.00		-1.20	
	E[CF]	0.6938±0.07%	0.6794±0.06%	0.6996±0.08%	0.6822±0.07%	0.6937±0.07%	0.6796±0.06%	0.6879±0.08%	0.6721±0.06%
	ΔE[CF]	–	-2.08	0.84	-1.67	-0.01	-2.05	-0.85	-3.13
0.8	CPUa	6212.82		8.26		2601.44		4.43	
	CPUp	0.02	0.77	0.00	0.00	0.01	0.82	0.00	0.00
	CPU	6212.84	6213.59	8.26	8.26	2601.45	2601.26	4.43	4.43
	R	1579027.6		1577056.3		1579027.6		1562902.7	
	ΔR	–		-0.12		0.00		-1.02	
	E[R]	1531298.4±0.06%	1506554.9±0.05%	1536923.4±0.06%	1509048.0±0.05%	1531943.5±0.06%	1506015.7±0.05%	1522267.3±0.06%	1496016.3±0.05%
	ΔE[R]	–	-1.62	0.37	-1.45	0.04	-1.65	-0.59	-2.30
	CF	0.8137		0.8134		0.8137		0.8089	
	ΔCF	–		-0.03		0.00		-0.59	
	E[CF]	0.7865±0.06%	0.7767±0.05%	0.7913±0.07%	0.7781±0.06%	0.7867±0.06%	0.7764±0.05%	0.7860±0.06%	0.7740±0.06%
	ΔE[CF]	–	-1.24	0.61	-1.06	0.03	-1.28	-0.06	-1.54
1.0	CPUa	5095.71		12.55		1583.37		5.37	
	CPUp	0.01	0.87	0.00	0.02	0.01	0.82	0.00	0.00
	CPU	5095.71	5095.71	12.55	12.57	1583.38	1583.19	5.37	5.37
	R	1771433.6		1767263.0		1771433.6		1753343.1	
	ΔR	–		-0.24		0.00		-1.02	
	E[R]	1714632.3±0.06%	1698282.4±0.05%	1718954.8±0.05%	1701311.1±0.05%	1716095.7±0.06%	1697962.3±0.05%	1704528.5±0.05%	1687988.5±0.05%
	ΔE[R]	–	-0.95	0.25	-0.78	0.09	-0.97	-0.59	-1.55
	CF	0.8735		0.8707		0.8735		0.8635	
	ΔCF	–		-0.32		0.00		-1.15	
	E[CF]	0.8423±0.06%	0.8383±0.05%	0.8461±0.06%	0.8387±0.05%	0.8431±0.06	0.8381±0.05%	0.8384±0.06%	0.8314±0.05%
	ΔE[CF]	–	-0.47	0.45	-0.44	0.09	-0.49	-0.46	-1.29
1.2	CPUa	4648.74		7.95		1692.23		4.99	
	CPUp	0.00	0.87	0.00	0.00	0.01	0.77	0.00	0.00
	CPU	4648.74	4649.62	7.95	7.95	1692.24	1693.00	4.99	4.99
	R	1925482.6		1921674.5		1925482.6		1908515.6	
	ΔR	–		-0.20		0.00		-0.88	
	E[R]	1862206.9±0.05%	1851161.2±0.04%	1868049.6±0.05%	1858369.4±0.04%	1863181.2±0.1%	1855135.9±0.0%	1853486.0±0.05%	1844254.3±0.05%
	ΔE[R]	–	-0.59	0.31	-0.21	0.05	-0.38	-0.47	-0.96
	CF	0.9028		0.9003		0.9028		0.8935	
	ΔCF	–		-0.28		0.00		-1.03	
	E[CF]	0.8700±0.06%	0.8687±0.05%	0.8753±0.05%	0.8711±0.04%	0.8701±0.05%	0.8685±0.05%	0.8676±0.05%	0.8638±0.05%
	ΔE[CF]	–	-0.14	0.61	0.12	0.01	-0.17	-0.27	-0.71
1.4	CPUa	6651.45		9.05		1800.12		5.52	
	CPUp	0.00	0.73	0.00	0.00	0.02	0.74	0.00	0.00
	CPU	6651.45	6652.18	9.05	9.05	1800.14	1800.86	5.52	5.52
	R	2058818.3		2054818.4		2058818.3		2038291.7	
	ΔR	–		-0.19		0.00		-1.00	
	E[R]	1991666.0±0.06%	1985537.7±0.04%	1996495.9±0.05%	1992704.9±0.04%	1992172.5±0.06%	1985049.3±0.05%	1979836.8±0.05%	1977170.6±0.04%
	ΔE[R]	–	-0.31	0.24	0.05	0.03	-0.33	-0.59	-0.73
	CF	0.9268		0.9231		0.9268		0.9157	
	ΔCF	–		-0.40		0.00		-1.20	
	E[CF]	0.8932±0.06%	0.8932±0.05%	0.8968±0.05%	0.8952±0.04%	0.8936±0.06%	0.8930±0.06%	0.8890±0.05%	0.8882±0.04%
	ΔE[CF]	–	-0.00	0.41	0.22	0.04	-0.02	-0.47	-0.56
	$\overline{\text{CPU}}$	5532.58	5533.17	8.73	8.74	1865.87	1866.10	4.83	4.83
	$\overline{\Delta\text{E[R]}}$	–	-1.18	0.29	-0.97	0.04	-1.15	-0.59	-1.72
	$\overline{\Delta\text{E[CF]}}$	–	-0.78	0.58	-0.57	0.03	-0.80	-0.42	-1.44

Each approximation is solved in **CPUa** seconds and return an optimal revenue **R** corresponding to a capacity factor **CF**. We then transform this solution to policy(ies). This transformation takes **CPUp** seconds and is then simulated in a discrete arrivals simulation with **500** evaluations to obtain an expected revenue **E[R]** and expected capacity factor **E[CF]** for a 95% confidence interval. The total running time is **CPU** and we calculate **ΔE[CF]** and **ΔE[R]** the capacity factor and expected revenue relative difference with respect to CDLP-OP.

ANNEXE B ARTICLE 2 E-COMPANION

B.1 Parallel flights

Instance is completely described at *https ://thibaultbarbier.com*.

Tableau B.1 Approximations results for the Parallel flights instance.

		CPUa				R				CF			
LF	$w_\emptyset$	CDLP	MBGP	CDBG	LBGP	CDLP	MBGP	CDBG	LBGP	CDLP	MBGP	CDBG	LBGP
0.8	0.01	0.25	0.14	0.08	0.04	35251.4	35251.4	35251.4	35251.4	0.78	0.78	0.78	0.78
	0.05	0.00	0.03	0.05	0.03	32538.5	32538.5	32538.5	32538.5	0.72	0.72	0.72	0.72
	0.1	0.02	0.08	0.06	0.02	29691.5	29691.5	29691.5	29691.5	0.66	0.66	0.66	0.66
1.0	0.01	0.49	0.10	0.19	0.02	43878.3	43523.9	43878.3	40642.4	0.98	0.97	0.98	0.90
	0.05	0.25	0.10	0.08	0.02	40671.2	40643.1	40671.2	40643.1	0.90	0.90	0.90	0.90
	0.1	0.00	0.02	0.03	0.02	37114.4	37114.4	37114.4	37114.4	0.82	0.82	0.82	0.82
1.2	0.01	0.00	0.10	0.11	0.03	45000.0	45000.0	45000.0	40642.4	1.00	1.00	1.00	0.90
	0.05	0.02	0.09	0.17	0.02	45000.0	45000.0	45000.0	40643.1	1.00	1.00	1.00	0.90
	0.1	0.16	0.03	0.15	0.03	43713.1	43713.1	43713.1	40642.4	1.00	1.00	1.00	0.90
Average		0.13	0.08	0.10	0.02	39206.5	39164.0	39206.5	37534.4	0.87	0.87	0.87	0.83

Tableau B.2 Approximation-policy results for the Parallel flights instance.

			CDLP			MBGP			CDBG			LBGP		
LF		$w_\emptyset$	OP	PB	OD	PC	PB	OD	OP	PB	OD	PC	PB	OD
0.8	CPU	0.01	0.00	0.00	9.39	0.00	0.00	10.37	0.00	0.00	10.83	0.00	0.00	10.78
		0.05	0.00	0.00	10.84	0.00	0.00	11.08	0.00	0.00	10.81	0.00	0.00	10.89
		0.1	0.00	0.00	10.88	0.00	0.00	11.70	0.00	0.00	11.61	0.00	0.00	11.90
	E[R]	0.01	35267.9±0.29	34216.4±0.17	35165.8±0.29	35292.9±0.29	34240.2±0.18	35270.9±0.29	35220.5±0.28	34194.8±0.18	35316.2±0.28	35228.9±0.28	34219.5±0.18	35273.0±0.29
		0.05	32538.2±0.30	31519.9±0.19	32479.3±0.30	32573.1±0.30	31616.7±0.19	32525.9±0.30	32522.7±0.29	31514.5±0.19	32495.7±0.30	32510.9±0.30	31555.9±0.19	32510.1±0.30
		0.1	29620.7±0.32	28714.8±0.21	29620.1±0.31	29726.3±0.32	28758.3±0.20	29725.6±0.31	29740.9±0.31	28697.0±0.20	29710.3±0.31	29673.4±0.32	28736.1±0.20	29701.3±0.32
	E[CF]	0.01	0.78±0.29	0.76±0.17	0.78±0.29	0.78±0.29	0.76±0.18	0.78±0.29	0.78±0.28	0.76±0.18	0.78±0.28	0.78±0.28	0.76±0.18	0.78±0.29
		0.05	0.72±0.30	0.70±0.19	0.72±0.30	0.72±0.30	0.70±0.19	0.72±0.30	0.72±0.29	0.70±0.19	0.72±0.30	0.72±0.30	0.70±0.19	0.72±0.30
		0.1	0.66±0.32	0.64±0.21	0.66±0.31	0.66±0.32	0.64±0.20	0.66±0.31	0.66±0.31	0.64±0.20	0.66±0.31	0.66±0.32	0.64±0.20	0.66±0.32
1.0	CPU	0.01	0.00	0.00	15.87	0.00	0.00	15.57	0.02	0.00	15.69	0.00	0.00	15.04
		0.05	0.00	0.00	14.20	0.00	0.00	14.18	0.00	0.00	14.30	0.00	0.00	14.17
		0.1	0.00	0.00	14.35	0.00	0.02	14.40	0.00	0.00	14.64	0.00	0.00	14.84
	E[R]	0.01	42764.6±0.17	42729.5±0.14	43019.5±0.18	42623.8±0.19	42585.7±0.14	42710.8±0.21	42705.5±0.18	42729.2±0.15	42979.5±0.18	40302.3±0.24	40659.8±0.07	42886.5±0.19
		0.05	40359.1±0.24	39533.3±0.15	40339.7±0.24	40389.1±0.24	39469.2±0.17	40445.3±0.26	40332.8±0.24	39478.5±0.16	40360.0±0.24	40320.3±0.23	39485.8±0.16	39534.1±0.26
		0.1	36984.0±0.27	35891.4±0.18	36978.0±0.28	36888.4±0.28	35973.3±0.18	37027.1±0.27	37080.1±0.27	36020.6±0.17	36988.8±0.27	37069.3±0.28	36006.5±0.18	37026.5±0.27
	E[CF]	0.01	0.95±0.17	0.95±0.14	0.96±0.18	0.95±0.19	0.95±0.14	0.95±0.21	0.95±0.18	0.95±0.15	0.96±0.18	0.90±0.24	0.90±0.07	0.95±0.19
		0.05	0.90±0.24	0.88±0.15	0.90±0.24	0.90±0.24	0.88±0.17	0.88±0.26	0.90±0.24	0.88±0.16	0.90±0.24	0.90±0.23	0.88±0.16	0.88±0.26
		0.1	0.82±0.27	0.80±0.18	0.82±0.28	0.82±0.28	0.80±0.18	0.82±0.27	0.82±0.27	0.80±0.17	0.82±0.27	0.82±0.28	0.80±0.18	0.82±0.27
1.2	CPU	0.01	0.00	0.00	18.72	0.00	0.00	18.77	0.00	0.00	18.43	0.00	0.00	19.20
		0.05	0.00	0.00	18.08	0.00	0.00	17.66	0.00	0.00	17.57	0.00	0.00	17.88
		0.1	0.00	0.00	17.57	0.00	0.00	17.48	0.00	0.00	17.47	0.00	0.00	17.58
	E[R]	0.01	43231.8±0.15	44964.8±0.02	44734.0±0.35	44533.6±0.15	44814.0±0.02	44059.0±0.35	43540.7±0.15	44963.3±0.02	44454.6±0.33	40382.3±0.24	41000.0±0.00	44767.4±0.02
		0.05	43337.0±0.16	44571.2±0.08	44964.8±0.35	43439.8±0.15	44601.5±0.07	44902.6±0.35	43364.3±0.16	44610.5±0.07	44880.2±0.34	40322.8±0.24	40838.2±0.01	43954.0±0.13
		0.1	42247.3±0.16	42581.4±0.15	42885.5±0.17	42276.4±0.16	42550.7±0.15	42827.0±0.17	42296.9±0.16	42534.8±0.15	42907.1±0.17	40250.0±0.24	40633.6±0.06	41198.7±0.22
	E[CF]	0.01	0.96±0.15	1.00±0.02	0.79±0.35	0.97±0.15	1.00±0.02	0.80±0.35	0.97±0.15	1.00±0.02	0.83±0.33	0.90±0.24	0.91±0.00	0.99±0.02
		0.05	0.96±0.16	0.99±0.08	0.80±0.35	0.97±0.15	0.99±0.07	0.80±0.35	0.96±0.16	0.99±0.07	0.80±0.34	0.90±0.24	0.91±0.01	0.98±0.13
		0.1	0.96±0.16	0.97±0.14	0.96±0.17	0.96±0.16	0.97±0.14	0.96±0.17	0.96±0.16	0.97±0.15	0.96±0.17	0.89±0.24	0.91±0.06	0.93±0.24

B.2 Bus-line

Instance is completely described at *https ://thibaultbarbier.com*.

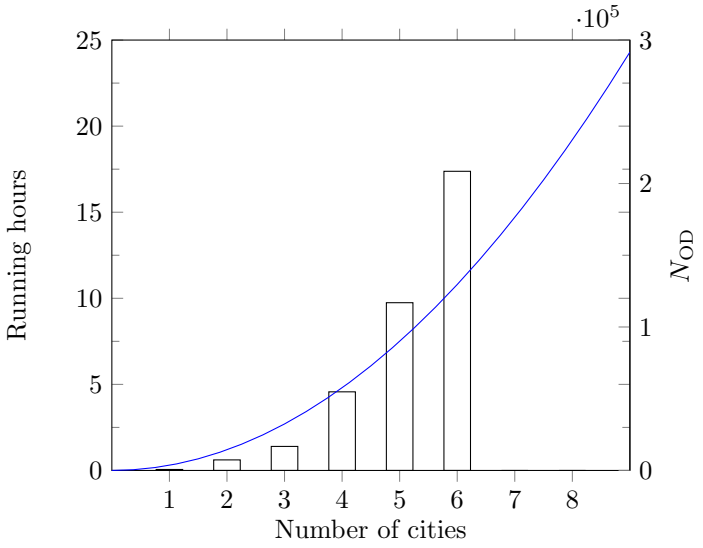

Figure B.1 Time needed to build the OD policy for the Bus-line ($LF = 1$, $w_\emptyset = 1\%$).

Tableau B.3 Approximations results for the Bus-line instance.

		CPU				R				CF			
LF	$w_\emptyset^{\%}$	CDLP	MBGP	CDBG	LBGP	CDLP	MBGP	CDBG	LBGP	CDLP	MBGP	CDBG	LBGP
0.8	1	65.18	94.57	44.30	1.97	69341.8	69451.9	69434.9	64807.1	0.71	0.71	0.71	0.64
	5	6.77	3.83	10.21	2.09	64327.8	64413.5	64413.5	63393.6	0.66	0.66	0.66	0.64
	10	7.71	1.38	6.56	1.43	59694.6	59915.7	59915.7	59915.7	0.62	0.62	0.62	0.62
1.0	1	35.47	2564.61	271.54	1.83	78791.7	78751.0	78776.3	72669.7	0.82	0.82	0.82	0.72
	5	6.93	10.86	13.06	1.78	74052.8	74197.5	74134.5	68833.3	0.78	0.78	0.78	0.69
	10	7.90	1.53	7.85	1.86	69980.1	70093.5	70093.7	70093.5	0.75	0.75	0.75	0.75
1.2	0.01	8.82	11424.58	5020.03	12.57	85072.0	85000.5	85066.8	78971.7	0.88	0.88	0.88	0.78
	0.05	11.41	367.27	72.37	2.41	81266.3	81331.3	81303.0	75187.0	0.85	0.85	0.85	0.75
	0.1	5.83	9.78	5.89	2.25	76874.8	77023.6	76950.6	71456.7	0.82	0.82	0.82	0.73
Average		17.33	1608.71	605.76	3.13	73266.9	73353.2	73343.2	69480.9	0.76	0.76	0.76	0.70

Tableau B.4 Approximation-policy results for the Bus-line instance.

LF		$w_{\emptyset}^{\%}$	CDLP		MBGP		CDBG		LBGP	
			OP	PB	PC	PB	OP	PB	PC	PB
0.8	CPU	1	0.02	0.02	0.00	0.02	0.00	0.00	0.00	0.02
		5	0.00	0.02	0.00	0.02	0.00	0.02	0.00	0.02
		10	0.00	0.02	0.00	0.00	0.00	0.02	0.00	0.00
	E[R]	1	66510.5±0.17	64268.2±0.14	66653.4±0.17	62462.2±0.13	66648.6±0.18	64165.5±0.14	62671.4±0.19	60481.3±0.14
		5	62074.4±0.19	60318.9±0.15	62229.7±0.19	58995.4±0.15	62133.9±0.19	60043.3±0.15	61214.5±0.20	58713.5±0.15
		10	57175.2±0.19	55653.5±0.15	57536.4±0.19	54037.6±0.15	57529.4±0.19	55408.4±0.15	57615.1±0.19	54104.8±0.15
	E[CF]	1	0.68±0.19	0.66±0.16	0.68±0.19	0.66±0.16	0.68±0.20	0.65±0.15	0.62±0.21	0.63±0.16
		5	0.64±0.22	0.62±0.18	0.64±0.22	0.62±0.17	0.64±0.23	0.61±0.18	0.62±0.23	0.61±0.17
		10	0.60±0.22	0.58±0.18	0.60±0.22	0.57±0.17	0.60±0.22	0.57±0.18	0.60±0.22	0.58±0.17
1.0	CPU	1	0.00	0.02	0.00	0.00	0.00	0.02	0.00	0.00
		5	0.00	0.02	0.00	0.02	0.02	0.02	0.00	0.02
		10	0.00	0.02	0.00	0.00	0.00	0.03	0.00	0.00
	E[R]	1	74183.4±0.14	73507.4±0.11	73955.0±0.14	71476.1±0.11	74103.8±0.14	73432.4±0.12	69579.3±0.16	67398.2±0.11
		5	70299.7±0.15	69146.7±0.12	70689.1±0.15	68454.4±0.12	70591.1±0.16	69455.4±0.12	65923.3±0.16	63659.0±0.12
		10	66462.7±0.16	64837.9±0.13	66581.1±0.16	63757.7±0.13	66530.3±0.15	65050.4±0.13	66650.8±0.16	63728.4±0.12
	E[CF]	1	0.77±0.15	0.76±0.13	0.76±0.15	0.76±0.13	0.76±0.15	0.76±0.14	0.69±0.17	0.70±0.12
		5	0.74±0.17	0.73±0.14	0.74±0.17	0.73±0.14	0.74±0.17	0.73±0.14	0.66±0.18	0.66±0.13
		10	0.71±0.18	0.69±0.15	0.71±0.18	0.70±0.15	0.71±0.17	0.69±0.15	0.71±0.18	0.70±0.14
1.2	CPU	1	0.00	0.02	0.01	0.01	0.00	0.02	0.00	0.00
		5	0.00	0.02	0.00	0.01	0.00	0.02	0.00	0.00
		10	0.00	0.03	0.00	0.00	0.00	0.02	0.00	0.00
	E[R]	1	79578.9±0.13	79521.4±0.10	79344.8±0.13	77703.3±0.10	79218.7±0.13	79544.3±0.10	74905.6±0.14	74627.3±0.10
		5	76215.1±0.13	76094.6±0.11	76457.2±0.13	73416.0±0.11	76433.2±0.13	75872.9±0.11	71651.2±0.16	69968.9±0.11
		10	72399.0±0.14	71729.6±0.11	72482.7±0.14	70549.6±0.11	72508.5±0.14	71942.8±0.11	68178.6±0.16	67182.6±0.11
	E[CF]	1	0.82±0.13	0.82±0.12	0.81±0.14	0.82±0.11	0.81±0.14	0.82±0.11	0.74±0.16	0.75±0.11
		5	0.80±0.14	0.80±0.12	0.80±0.14	0.79±0.12	0.80±0.14	0.79±0.13	0.71±0.17	0.72±0.12
		10	0.78±0.15	0.77±0.13	0.77±0.15	0.77±0.13	0.78±0.15	0.77±0.13	0.69±17	0.70±0.12

B.3 Airline

Instance is completely described at *http ://thibaultbarbier.com*.

ANNEXE C ARTICLE 3 APPENDIX

All instances are completely described at *https ://thibaultbarbier.com*.

Tableau C.1 Optimal policies for both instances.

	LF	FAS			$DAS_{5\%}$					$DAS_{1\%}$					$DAS_{0.5\%}$				
		μ	CPU	CRS	μ	CPU	CRS	N	σ	μ	CPU	CRS	N	σ	μ	CPU	CRS	N	σ
Bus-line	0.8	92600.8	0.124	22.0	85533.37	0.296	944.9	10.0	10209287.0	87450.81	2.294	962.2	126.0	6247656.2	87304.62	9.612	961.9	557.0	6895921.2
	1.0	102771.8	0.048	28.0	95450.90	0.172	1205.6	9.0	11827927.0	95786.65	1.638	1206.2	82.0	4884783.1	95503.05	7.880	1199.6	401.0	5931943.5
	1.2	110140.9	0.031	30.0	102935.03	0.110	1439.3	4.0	5376764.4	102537.98	1.653	1439.9	74.0	5062021.2	102723.51	8.139	1438.7	360.0	6149063.3
Airline	0.8	1577010.7	2.902	76.0	1532008.23	13.603	14788.8	4.0	94893130.2	1532144.54	16.224	14669.8	5.0	66112325.5	1536161.05	102.131	14732.5	31.0	113502139.7
	1.0	1767257.4	2.574	87.0	1726482.23	14.530	18477.3	4.0	144125012.0	1718816.66	24.695	18459.0	7.0	121457926.4	1717689.62	74.958	18348.2	21.0	98069564.2
	1.2	1921637.3	2.668	96.0	1857434.60	17.004	21991.3	4.0	237029724.3	1873314.53	16.792	22201.3	4.0	73871673.7	1872668.46	38.686	22072.3	9.0	48479595.2

Tableau C.2 Bus-line with 150 random policies per PC scenario.

LF	Close	FAS			$DAS_{5\%}$					$DAS_{1\%}$					$DAS_{0.5\%}$				
		μ	CPU	CRS	μ	CPU	CRS	N	σ	μ	CPU	CRS	N	σ	μ	CPU	CRS	N	σ
	0	56537.3	0.102	101.3	56094.36	0.107	962.1	7.0	2921638.5	56018.76	2.854	960.1	188.1	3812418.5	56013.19	11.377	960.0	759.1	3858877.2
	10	56104.2	0.090	96.7	55703.74	0.115	962.3	7.2	3007110.8	55620.41	2.937	960.0	192.9	3858683.5	55619.21	11.861	960.0	782.0	3925145.6
	20	55555.9	0.074	91.2	55010.18	0.108	960.5	7.7	3232646.2	55044.11	2.866	959.4	204.2	4004469.6	55039.72	11.667	959.9	830.7	4078813.4
	30	53931.2	0.064	86.1	53458.60	0.110	962.0	8.4	3300121.4	53482.54	2.960	959.5	222.3	4110774.5	53492.86	11.995	959.8	903.6	4194560.8
	40	52847.9	0.055	79.8	52297.73	0.098	960.9	8.0	3118408.1	52380.90	2.701	959.6	219.4	3894023.5	52381.77	11.442	959.9	926.1	4117838.1
0.8	50	50049.1	0.043	70.7	49637.00	0.099	961.0	8.9	3188065.2	49635.22	2.791	959.9	249.9	3981737.4	49634.73	12.502	960.0	1005.1	4018042.3
	60	46558.0	0.032	60.7	46202.32	0.101	959.4	9.4	2902915.1	46172.58	3.136	960.2	287.2	3957792.2	46173.27	12.682	959.9	1152.1	3970582.2
	70	41428.5	0.024	50.0	41103.90	0.118	959.6	12.4	3153471.2	41083.03	3.291	959.9	337.2	3664334.4	41067.55	13.091	960.0	1343.6	3655435.1
	80	34478.5	0.012	37.2	34066.80	0.143	961.1	16.5	2856063.1	34138.44	3.641	960.4	460.7	3387334.5	34114.18	14.408	960.0	1849.8	3407358.0
	90	22142.2	0.004	21.7	21950.56	0.193	959.4	31.3	2151097.9	21968.58	5.725	960.1	913.4	2610901.6	21957.67	23.164	960.0	3693.9	2635124.2
	100	0.0	0.000	1.0	0.00	0.017	967.2	4.0	0.0	0.00	0.016	963.5	4.0	0.0	0.00	0.016	957.4	4.0	0.0
	0	62476.1	0.089	92.1	61726.12	0.105	1198.4	5.8	2754979.4	61843.55	2.333	1199.3	128.3	3168554.3	61836.65	9.426	1200.0	517.8	3217196.1
	10	61699.5	0.080	89.5	61105.22	0.106	1201.6	6.3	3028752.7	61124.78	2.449	1200.4	143.0	3453028.6	61108.19	9.541	1199.9	557.0	3374719.0
	20	61508.9	0.069	85.2	60817.15	0.093	1200.6	5.9	2663804.9	60799.68	2.348	1199.5	146.1	3490234.7	60821.31	9.474	1199.8	588.5	3532049.3
	30	60340.5	0.061	81.4	59611.68	0.091	1200.5	6.0	2785956.9	59707.01	2.432	1199.7	160.0	3684424.6	59703.52	9.711	1200.0	638.2	3687769.0
	40	58718.4	0.050	73.6	58071.25	0.101	1198.4	6.9	3011886.0	58087.20	2.523	1200.3	171.8	3719482.8	58056.05	10.123	1199.8	687.2	3735746.0
1.0	50	55466.6	0.040	67.7	54441.24	0.065	1202.3	4.8	1576191.8	54946.28	2.801	1200.0	205.9	4036720.4	54933.19	11.448	1200.1	839.5	4101627.4
	60	53099.8	0.032	60.9	52783.24	0.094	1199.3	7.5	2564536.5	52648.35	2.176	1200.3	172.6	3074711.0	52687.85	10.644	1200.6	843.4	3783906.9
	70	45590.0	0.022	51.6	45054.71	0.163	1199.4	13.4	4023351.7	45168.85	3.945	1200.0	321.4	4199301.3	45153.91	15.659	1200.6	1275.1	4182286.1
	80	35287.2	0.010	34.8	35017.10	0.185	1198.8	18.3	3178027.4	35133.32	4.452	1201.0	435.0	3380377.7	35058.14	18.080	1200.1	1765.4	3413271.8
	90	24203.1	0.004	21.7	24041.96	0.289	1204.6	31.8	2887619.8	23988.94	7.199	1200.4	784.8	2875117.7	23977.68	28.598	1200.2	3114.5	2865961.3
	100	0.0	0.000	1.0	0.00	0.025	1204.9	4.0	0.0	0.00	0.026	1207.0	4.0	0.0	0.00	0.026	1201.3	4.0	0.0
	0	66206.9	0.084	83.7	65604.91	0.101	1443.9	5.0	2125171.0	65780.12	2.216	1439.3	108.9	3039319.2	65841.30	8.206	1440.8	403.2	2841232.4
	10	65797.6	0.076	82.9	65872.43	0.121	1439.6	6.5	3572049.2	65312.83	2.206	1440.9	115.8	3179839.0	65216.81	8.452	1440.2	442.8	3046955.7
	20	65459.0	0.067	80.9	64778.36	0.106	1445.1	5.7	2751926.2	64798.86	1.895	1442.0	102.8	2798014.8	64791.50	8.503	1439.7	462.0	3148926.7
	30	64675.1	0.057	74.7	64406.83	0.095	1447.8	5.5	2523237.3	64009.93	2.273	1437.9	130.0	3438067.8	64074.90	9.108	1439.9	519.2	3454938.4
	40	63326.3	0.049	71.3	62378.14	0.099	1440.0	5.8	3002225.1	62721.75	2.436	1439.9	144.3	3679479.8	62734.53	10.161	1439.8	598.9	3830514.7
1.2	50	60784.2	0.039	63.4	60334.00	0.122	1446.5	7.8	4304230.0	60188.26	2.132	1441.4	137.5	3229970.5	60151.86	8.903	1440.2	572.1	3376911.2
	60	57095.5	0.030	58.0	56376.70	0.096	1442.6	6.6	2721236.4	56601.59	2.918	1441.1	197.9	4113755.1	56656.19	11.742	1440.0	796.6	4147954.2
	70	48037.9	0.018	45.2	47758.24	0.142	1441.5	10.5	3799538.8	47740.32	3.333	1440.4	248.8	3707782.6	47792.62	13.236	1440.4	988.7	3646478.0
	80	41793.2	0.010	35.0	41548.49	0.146	1438.5	12.0	2934531.4	41525.58	4.054	1441.0	340.9	3611829.0	41505.62	16.075	1439.5	1347.9	3593111.0
	90	28048.0	0.003	20.1	27915.51	0.211	1435.6	21.3	2443094.4	27860.12	5.365	1440.0	527.4	2576515.6	27870.81	21.771	1440.3	2139.3	2625548.8
	100	0.0	0.000	1.0	0.00	0.032	1441.6	4.0	0.0	0.00	0.030	1443.7	4.0	0.0	0.00	0.028	1434.4	4.0	0.0

Tableau C.3 Airline with 50 random policies per PC scenario.

LF	Close	FAS			$DAS_{5\%}$					$DAS_{1\%}$					$DAS_{0.5\%}$				
		μ	CPU	CRS	μ	CPU	CRS	N	σ	μ	CPU	CRS	N	σ	μ	CPU	CRS	N	σ
	0	1036884.9	6.158	309.8	1035345.63	7.956	14723.9	4.0	95297381.6	1035097.75	18.037	14734.7	9.1	55766726.6	1035026.42	70.425	14733.8	35.5	59860729.8
	10	1012809.5	5.528	300.1	1011513.38	7.920	14749.4	4.0	72285052.6	1011869.55	17.791	14746.3	9.0	52324969.9	1011821.45	79.087	14733.7	39.9	65366514.4
	20	994411.4	5.036	287.4	994189.12	7.553	14725.0	4.0	70366851.1	993855.04	15.366	14747.0	8.1	45966566.6	994389.44	75.610	14742.3	40.1	62598393.6
	30	977374.5	4.348	267.8	974643.35	7.296	14746.3	4.0	81224022.4	975654.57	18.644	14727.6	10.3	56191646.7	975952.92	76.686	14730.9	42.2	63815236.3
	40	932862.6	3.759	253.8	931963.23	6.799	14733.2	4.0	70582000.9	931074.57	20.587	14738.1	12.3	63999581.8	931470.17	83.525	14732.9	50.1	69123216.5
0.8	50	888975.3	2.914	232.5	887217.33	6.127	14737.9	4.0	64117001.0	886392.97	17.140	14726.6	11.2	51859982.3	886970.68	76.340	14730.6	49.8	62434919.6
	60	842022.9	2.304	212.1	841742.04	6.018	14742.1	4.0	64244840.8	840035.83	17.868	14735.8	11.9	50421832.9	840241.18	72.074	14728.0	48.1	53940928.5
	70	767356.8	1.650	182.4	765198.97	5.598	14734.1	4.0	42989441.7	764863.24	19.458	14730.5	14.0	49074125.4	765978.79	87.671	14734.7	62.6	58943370.0
	80	677436.6	0.998	152.0	677084.15	5.655	14719.7	4.0	52735082.5	675961.69	21.888	14735.4	15.5	42905672.2	676054.41	100.622	14730.2	70.7	52082792.6
	90	479920.2	0.410	108.0	477958.09	5.936	14728.1	4.0	41420370.6	479085.50	47.909	14746.4	32.2	46516668.2	478665.06	204.576	14735.8	137.6	50230505.1
	100	0.0	0.000	1.0	0.00	3.966	14738.3	4.0	0.0	0.00	3.970	14747.9	4.0	0.0	0.00	3.965	14744.2	4.0	0.0
	0	1151504.4	5.584	285.0	1148955.20	8.739	18403.7	4.0	79943389.2	1149083.74	16.465	18413.1	7.6	55904820.3	1149584.53	69.860	18411.5	32.0	66925193.9
	10	1134115.7	4.913	270.4	1132993.08	8.728	18413.2	4.0	83204396.3	1131246.66	16.501	18408.2	7.6	53782468.4	1132662.67	60.276	18417.1	28.1	56346434.6
	20	1109327.5	4.381	253.1	1108683.29	8.053	18408.6	4.0	51590047.9	1108918.89	18.569	18398.4	9.2	65707270.8	1108365.85	59.192	18419.9	29.3	57339959.2
	30	1081099.5	3.879	244.6	1079690.95	7.705	18444.1	4.0	66725733.3	1079376.48	12.354	18420.4	6.4	39441581.6	1081067.17	54.131	18434.1	28.0	51021333.5
	40	1032990.5	3.215	222.5	1030358.74	7.253	18409.4	4.0	69253853.8	1031280.26	16.560	18404.3	9.1	55029788.5	1032217.19	71.920	18433.2	39.6	67667746.4
1.0	50	991126.5	2.668	217.0	991087.37	6.746	18402.5	4.0	74623096.6	987777.64	18.496	18405.3	11.0	62132748.0	989075.00	63.672	18421.1	37.6	58647366.4
	60	926371.6	1.998	190.2	925409.40	6.716	18405.0	4.0	75171115.0	925650.33	20.050	18423.1	11.8	61010298.7	925020.86	73.316	18408.0	43.4	59392005.2
	70	860785.5	1.532	173.2	857986.10	6.382	18404.8	4.0	53207193.8	859793.50	18.488	18438.9	11.7	52661987.5	858800.70	85.522	18409.7	53.9	62340181.9
	80	732914.0	0.925	143.8	731382.05	6.450	18414.3	4.0	59583522.6	730700.18	22.015	18416.3	13.5	42488674.5	731824.68	91.062	18421.7	55.5	47792181.0
	90	538481.3	0.361	96.7	536110.18	6.801	18422.6	4.0	46290822.8	535763.02	30.602	18412.4	18.0	31735035.1	537460.67	162.564	18420.0	95.9	44038911.1
	100	0.0	0.000	1.0	0.00	5.274	18409.7	4.0	0.0	0.00	5.267	18417.1	4.0	0.0	0.00	5.261	18403.8	4.0	0.0
	0	1238896.0	4.964	256.4	1236474.39	8.911	22120.9	4.0	120645725.6	1238156.51	13.820	22076.0	6.2	50926248.3	1236286.70	54.377	22101.6	24.4	58061849.3
	10	1220504.6	4.516	252.7	1220107.06	8.851	22145.0	4.0	45560197.4	1219042.73	18.828	22088.8	8.5	75020619.0	1219110.12	52.301	22105.4	23.6	53667753.7
	20	1190876.9	3.974	237.2	1188082.03	8.538	22091.8	4.0	52246091.4	1189150.21	11.093	22056.2	5.2	37155266.2	1188819.06	58.742	22092.1	27.5	60523862.6
	30	1171343.4	3.554	228.0	1170013.16	8.158	22116.7	4.0	76646075.7	1168867.33	14.720	22060.3	7.2	51118588.7	1170273.27	65.820	22111.2	32.2	69501307.5
	40	1104690.7	2.964	211.1	1102947.04	7.705	22101.9	4.0	62783463.0	1102604.45	16.188	22086.7	8.3	59356321.6	1103385.73	52.615	22112.7	27.4	52884161.6
1.2	50	1066002.2	2.297	193.5	1065033.98	7.218	22117.6	4.0	90991984.2	1064902.13	15.370	22107.2	8.5	56793062.7	1064411.04	46.662	22090.4	25.7	46625084.1
	60	998682.2	1.781	176.7	995694.51	7.209	22083.8	4.0	63031155.5	996957.72	14.519	22094.6	8.0	46524614.0	997061.59	76.815	22098.0	42.3	66928650.8
	70	931859.5	1.378	160.3	930049.79	6.727	22129.7	4.0	55157355.8	930090.75	12.410	22090.6	7.3	32348401.4	930538.88	72.315	22095.7	42.7	58440979.8
	80	809614.2	0.865	133.7	807634.25	7.069	22064.2	4.0	67003002.3	808300.85	19.695	22093.2	11.1	45431471.6	807825.92	92.551	22096.1	52.0	54188822.0
	90	595283.1	0.356	94.4	596550.86	7.610	22123.6	4.0	43202173.9	593057.34	36.838	22100.0	19.3	41433174.1	593760.43	164.953	22110.7	89.3	50322242.5
	100	0.0	0.000	1.0	0.00	5.917	22087.3	4.0	0.0	0.00	5.914	22110.2	4.0	0.0	0.00	5.934	22130.4	4.0	0.0